Using Multitask Learning to Understand Language Processing in the Brain

Dan Schwartz

CMU-LTI-20-006
August 2020

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Tom Mitchell, *(Chair)*, Carnegie Mellon University
Leila Wehbe, Carnegie Mellon University
Bhiksha Raj, Carnegie Mellon University
Stefan Frank, Radboud University Nijmegen

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies

Abstract

Understanding the cognitive processes involved in human language comprehension has been a longstanding goal in the scientific community. While significant progress towards that goal has been made, the processes involved in integrating a sequence of individual word meanings into the meaning of a clause, sentence, or discourse is poorly understood. Recently, the natural language processing (NLP) community has demonstrated that deep language models are, to an extent, capable of representing word meanings and performing integration of those meanings into a representation that can successfully capture the meaning of a sequence. In this thesis, we therefore leverage deep language models as an analysis tool to improve our understanding of human language processing. In the setting of multitask learning, we can gain insight into the mechanisms that deep language models use to make their predictions by comparing tasks to each other. Furthermore, if some of the task predictions we ask the model to make are relevant to cognitive processing — for example the prediction of eye-tracking data measured as participants read sentences — we can ultimately use those insights to better understand language processing in people. In this work, we first examine the use of constructive interference in multitask learning as an analysis tool. Constructive interference occurs when two tasks are related and a model is constrained by having to accurately predict both. In those cases, the representation the model learns often generalizes better to predicting unseen data than if that model had been trained on just one of those tasks. This is because the constraint that the representations must work for the prediction of both tasks provides a helpful inductive bias. If generalization error improves when tasks are trained together, this can be viewed, with caveats, as an indication that the tasks are related. In our experiments improved generalization error suggests relationships between event-related potential components (ERPs) that are consistent with the existing literature, and suggests other relationships which bear on the interpretation of ERPs. Next, we investigate what a deep language model learns when it is trained to predict brain activity recordings. We find that the information encoded into the parameters of the model helps the model predict brain activity, generalizes to prediction of unseen participants' brain activity, and to some degree, generalizes across different brain activity recording modalities. These findings provide evidence that the information the model encodes into its parameters is relevant to the cognitive processes underlying brain activity, and not just to idiosyncrasies in the data, making fine-tuning and multitask learning valid tools for probing those cognitive processes. Finally, we develop an analysis method in which a model learns a small number of latent functions which take a sequence of words as input and produce a representation from which multiple task outputs must be predicted. We assess task similarities based on the weights which map from the common latent representation to the output associated with each task. The similarities produced by this method capture expected relationships between NLP tasks, and can help us understand how a deep language model makes its predictions. We also examine the similarities between cognition-relevant tasks and NLP tasks, and find that the mechanisms underlying the model's predictions in cognition-relevant tasks are related to agent-like and patient-like semantic properties and to modifiers in a sentence. The methods developed here can be applied with different sets of tasks to gain different kinds of insight into both deep language models and cognitive processing, and offer a promising direction for understanding language processing in the brain.

Acknowledgments

I would like to thank my advisor, Tom Mitchell, for taking a chance on me and giving me the opportunity to study machine learning and computational neuroscience. It has been a fascinating journey and I'm grateful to have been able to take it. I have learned a lot over the course of my graduate studies, and Tom patiently allowed me to pursue many lines of research.

I would also like to thank Leila Wehbe for serving on my thesis committee, and especially for her encouragement and mentorship both when she was a student and when she returned to CMU as faculty. Leila has been a great collaborator, and her enthusiasm and ideas have been critical to my successes as a graduate student.

I am very grateful to Bhiksha Raj for serving on my committee, and for his support throughout my time at CMU. Bhiksha has consistently given me different perspectives on problems and has always been eager to talk, give guidance, and cheer on my work.

Thank you to Stefan Frank for taking the time to evaluate my thesis. Stefan's work over the years on using statistical models to evaluate ideas about cognition has served as an inspiration for some of the ideas in this work.

Thank you to Erika Laing and Dan Howarth for their contributions to data collection, experiment design, and preprocessing.

I would like to thank all of the students who have made my time at CMU special: Calvin Murdock and Nicole Rafidi for their friendship and taking me in as an unsanctioned officemate, and to Aaditya Ramdas, Anthony Platanios, Avinava Dubey, Ben Cowley, Christoph Dann, Dallas Card, Jesse Dodge, Maruan Al-Shedivat, Mrinmaya Sachan, Otilia Stretcu, and Willie Neisswanger for their friendship and for keeping life fun.

I would especially like to thank Mariya Toneva, who has been a great friend and collaborator. She had many ideas that have been incorporated into the work in this thesis in both formal contributions to chapter 4 and in countless helpful discussions we have had together. Without her friendship, help, and encouragement, this thesis could not have happened.

I would like to thank my mom, dad, and brother for their support throughout my life, for encouraging me to take a risk and pursue a PhD, and for cheering me on from the sidelines.

Finally I would like to thank Lacey. The last five years with her have been very special. She makes everyday better, and has supported me in many ways for which I am grateful. I look forward to many more happy days with her.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis we aim to use systems developed for natural language processing (NLP) in computers to better understand language processing in the brain. To achieve this end, we use multitask learning. While in a standard machine learning problem, a model estimates the relationship between one or more input variables and a single output variable, in multitask learning, a model estimates the relationship between one or more input variables and many output variables. Multitask learning has several properties that make it attractive as an analysis tool in the context of studying language processing in the brain. First, it enables us to combine together multiple complementary sources of information about language, including behavioral data, neural data, and data from natural language processing. Second, in our setup, we can combine together heterogeneous datasets, allowing us to learn from both small studies of human language processing and large natural language corpora even those datasets have no examples in common. Third, the inclusion of multiple tasks during training induces a bias that can be helpful for learning since each additional task further constrains the representations that the model uses. Fourth, the multiple tasks can be compared to one another in terms of either the constructive/destructive interference between tasks that occurs during learning or other measures of how similar a model believes the tasks are to each other. These task similarities, when used to compare a neural data prediction task to more interpretable tasks, such as part-of-speech prediction, enable a novel kind of interpretation of the underlying cognitive processes driving the brain activity in the neural data. In this thesis, we explore several variations of multitask learning on brain activity and show the effectiveness of transfer learning in this setting. We also use the framework to make some inferences about underlying language processing in the brain.

1.1 Motivation

While a great deal of progress has been made towards the goal of understanding how language processing works in the brain, parts of the language system are still poorly understood. Theories have been developed which broadly characterize lower level processes in the speech comprehension system (Hickok and Poeppel, 2007), and which characterize how words are mapped onto meaning (Lambon Ralph et al., 2010). A fair amount of evidence supports the idea that the sounds or glyphs which make up spoken and read words are mapped onto a widely distributed network of semantics in the brain, with visual aspects of meaning stored in visual areas, functional aspects stored in functional areas, and so on (see chapter 2). It is also thought that these various aspects of meaning are integrated into a single coherent representation for the meaning of a word by specific areas of the brain, but this process of integration is not very well understood. Higher levels of meaning integration, i.e. how the meanings of multiple different words are integrated together to form the meanings of phrases, sentences, and passages are even less well understood. At the

same time, in the last few years, the natural language processing (NLP) community has made substantial progress towards language understanding. Neural network models have reached parity with humans on certain language understanding benchmarks such as the general language understanding evaluation (GLUE) benchmark (Wang, Singh, et al., 2018; Wang, Pruksachatkun, et al., 2019). These benchmarks include tasks like question answering, understanding whether movie reviews are positive or negative, and predicting whether one sentence entails another. In order for a model to perform these tasks, it must, to some degree, be using representations which result from the higher levels of meaning integration we would like to study in people, i.e. integrating the meanings of words into a single representation of the meaning of a sentence or passage. In this thesis, we would like to leverage these advances from the natural language processing community to better understand the cognitive processes subserving language in the brain.

1.2 Approach

Our approach combines systems trained purely on language with datasets describing human behaviors and neural activity. This combination yields a model of human language processing which bears some resemblance to a computational simulation of human language processing, with important distinctions. Our approach leverages large and diverse datasets, powerful deep models of language, and modern optimization to guide the language representations and transformations used. The model in our approach is not constrained by biological considerations and may not perform computations in the same way as the human brain. However, in training the model to predict human behaviors and neural activity, which we sometimes refer to as cognition-relevant data, we encourage it to encode information that is similar to the information driving those data. In chapter 4, we validate those assertions, providing evidence that the information a model learns to encode when fine-tuned to predict neural activity is relevant to cognitive processes. We can therefore learn about the representations involved in human language processing by probing the information in a model trained to predict cognition-relevant data.

Multitask learning

In a multitask learning problem, we are given a set of tasks $\mathcal{T}$, where each task is itself a set of N_t example pairs. Letting the subscript t, i indicate the ith example within the tth task, we can write this as $\mathcal{S}_t = \left\{ \left(x_{t,1}, y_{t,1}^{(t)} \right), \left(x_{t,2}, y_{t,2}^{(t)} \right), ..., \left(x_{t,N_t}, y_{t,N_t}^{(t)} \right) \right\}$. Each $x_{t,i}$ in each task is treated as the value of a random variable X, or in a multivariate setting, as the multivariate value of a vector of random variables which we also denote as X. X is the same random variable (or same vector of random variables) across all of the tasks. In contrast, each $y_i^{(t)}$ is treated as the value of a random variable (or vector of random variables) $Y^{(t)}$ where that random variable (or vector of random variables) is task-specific. The goal in multitask learning is to jointly model the relationship between X and every $Y^{(t)}$.

General architectural scheme

We use a single deep network to make predictions about neural activity, behavior, and/or standard natural language processing tasks by treating each one as a task in a multitask learning framework, enabling the network to learn from all of them. In general, we aim to make the parts of the network that are specific to a single task as minimal as possible so that in the shared representations of the network we have a stronger bias to generalize across these tasks (Caruana, 1997). At a high level, our architectural scheme uses an encoder-decoder architecture. In such an architecture, an imaginary line is drawn which separates the model into two conceptual parts. The encoder is the part of the model which transforms the raw input

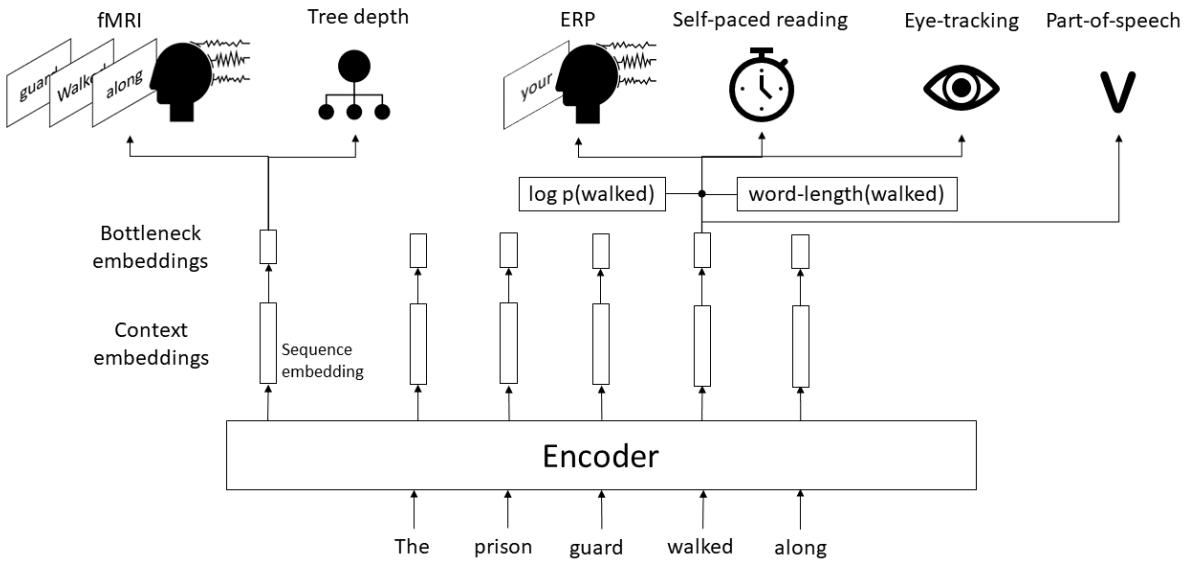


Figure 1.1: This sketch illustrates the key components of the architecture we use in our approach. We use a shared encoder, pre-trained as a language model on a large corpus. The outputs from this encoder in general go through a bottleneck in our architecture. In chapter 3, the bottleneck takes the form of an embedding of each pair of tokens, and the same embedding is shared across all tasks. In chapter 5, the bottleneck embedding is at the token level, and only minimal pooling is done when necessary before predictions are made from this shared embedding. In chapter 4, where the emphasis is not on a multitask setup, we do not use a bottleneck. We predict a mix of sequence-level, word-level, behavioral, neural, and language tasks. Although in this sketch it appears that all tasks are predicted for every example, in reality only a subset of the tasks are predicted on any given example.

into an intermediate form which captures the information that is pertinent to the multiple output tasks, and which is easier for the decoder(s) to use than the raw input. In a multitask problem, the part of the model which is common to all tasks is typically thought of as the encoder. The decoder (or decoders) are the part(s) of the model which transform the intermediate output produced by the encoder into the final output(s) that the model is trying to predict. In our specific case, the encoder is pre-trained on large corpora in a self-supervised setting to predict words that have been masked out of its input sequence or to predict the word following (or in some cases preceding) its input sequence — i.e. a language modeling task. We then use a fine-tuning stage of training during which several simple decoders (one per task) are trained. Each decoder consumes the outputs of the encoder and makes task-specific predictions. As the decoders are trained, we also continue training the encoder. We frequently also introduce a shared or partially shared bottleneck layer between the outputs of the encoder and the decoders (see Figure 1.1). The bottleneck can help to regularize the model, but more importantly it plays a crucial role in forcing the model to use similar representations for similar tasks.

From models to understanding

It is worth considering what we can expect to achieve from this endeavor. After all, while deep language models, such as BERT (Devlin et al., 2018), are able to predict neural activity to some extent, that alone does not improve our understanding of language processing in the brain. Even if the neural activity could be predicted perfectly by a language model, we would still need to probe that model in some way to gain insight into what information it uses to make those predictions. Interpretation of deep neural networks is an active field of research, and for the most part deep network models remain black box functions that map from their input to their output. However, many techniques for interpretation exist and we discuss some of the most relevant ones here, along with our own approach to interpretation. We believe that the task similarity approach we use can potentially lead to valuable insights about the cognitive processes in the brain.

Linking functions. One technique for interpretation that has been popular in the study of language processing in the brain is to use a linking function (see e.g. (Brennan, Stabler, et al., 2016; Hale et al., 2018; Frank, Otten, et al., 2015)). Linking functions are functions which reduce a model representation to a somewhat interpretable aspect of information in a model. For example, surprisal (the negative log probability) of a word in context according to the probability distribution computed by a language model has been used as a linking function from which the N400 response can be predicted (see chapter 3). To the extent that we can predict neural activity from these interpretable representations we learn about the latent cognitive processes that drive that activity. However, the interpretability comes at a cost: much of the predictive power from the underlying model is lost.

Prediction probes. A different type of technique looks to identify what information is available in the output (or at other layers) of a model, using classification tasks as probes (Adi et al., 2016; Linzen, Dupoux, and Goldberg, 2016) to see what can be decoded from the representations. Prediction probes quantify to what extent a specific kind of information is represented in a given layer, and given enough probes, the representation can be characterized in those terms.

Representational similarity analysis. Representational similarity analysis (RSA) (Kriegeskorte, Mur, and Bandettini, 2008; Shepard, 1957) can be used to quantify the similarity between different representations. In RSA, distance matrices are formed between each sample's representation and each other sample's representation using the representations determined by a given system (for example a model's representation of a sample, or the voxel activations for a stimulus in fMRI), similar to a kernel in machine learning. The distance matrices produced by two different systems' representations can then be compared to each other to determine how similar the representations are in the two systems. In application to deep networks, typically one distance matrix would be formed by using a layer of a network as a representation for a sample, and this would be compared to either a region of the brain, or to an interpretable space such as a space of hand designed features. Both prediction probes and RSA attempt to quantify what information is represented in a given layer, but in RSA, neither of the representation spaces necessarily has to be interpretable. For example, Yamins et al. (2014) uses RSA to compare neural network representations to representations in inferior temporal lobe (a high order vision area).

Ablation studies, occlusion studies, and gradient based techniques. Interpretation has sometimes been performed using ablation studies (also sometimes called lesioning), in which the contributions of various parts of a model to a prediction are assessed by removing or "damaging" those parts of the model

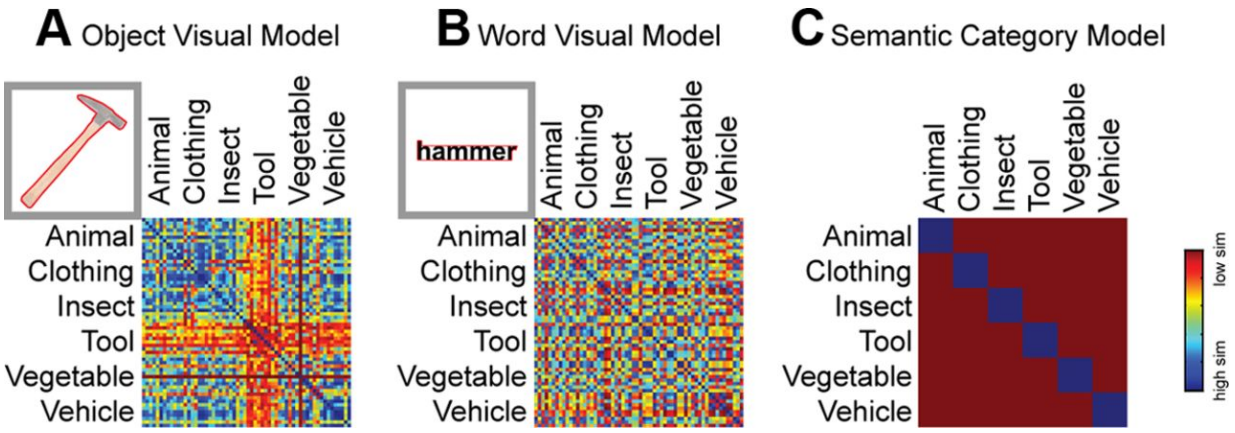


Figure 1.2: An example of RSA. Here we see the dissimilarity matrices for three different feature spaces on the same stimuli, which are, in this case, compared to a similar dissimilarity matrix derived from the fMRI images of regions of interest recorded while participants viewed picture and word stimuli (Devereux et al., 2013).

(see e.g. (Plaut, McClelland, et al., 1996; Girshick et al., 2014)). A related technique is to modify the input to a learned model at test time to identify the aspects of the input to which a model is sensitive using either rules and masking (Khandelwal et al., 2018; Jain and Huth, 2018; Toneva and Wehbe, 2019) or optimization of the input to activate specific nodes in the network (Erhan et al., 2009; Olah, Mordvintsev, and Schubert, 2017) (the latter is less common in language modeling). These techniques offer interpretations of what aspects of the input, an intermediate representation, or the function itself are important for the success of the model predictions.

Task similarity (our approach). All of these various techniques have advantages and disadvantages, and each can be used to learn something about what a model encodes and how it makes predictions. Indeed, one of the advantages of training a model to predict brain activity is that if a model is sufficiently accurate it can then be used as a surrogate for the brain activity itself, enabling probing which could not be done on the brain. Our architecture and multitask approach to learning create additional opportunities for interpretation. In chapter 3 we rely on (constructive) task interference as a way to measure whether different event-related potentials are related to each other and to eye-tracking and self-paced reading time data. Though imperfect (see (Caruana, 1997)), task interference can be viewed as measuring how well the outputs related to one task can be predicted when the model is constrained to also predict the outputs of other tasks well, and we interpret this as showing that there exists a representation which can capture the information from all of the relevant tasks. In chapter 5, rather than using task interference, we train a single model to predict all tasks, then compare the model parameters specific to each task to the model parameters specific to each other task to quantify task similarity. The task similarity approach bears some resemblance to RSA, but while RSA requires matched examples between the two systems (e.g. the brain and a computer vision model) being compared (in order to create comparable distance matrices), our approach compares model parameters/functions to each other, and thus enables comparison across heterogeneous datasets.

1.3 Research Questions

Methodological questions. This thesis sets out to answer questions about how deep language models can be used to improve our understanding of language processing in the brain. Many other researchers have also become interested in this question (Jain and Huth, 2018; Kell et al., 2018; Wehbe, Vaswani, et al., 2014), with some in particular trying to adapt techniques from the community that relates deep computer vision models to the visual system in the brain (Yamins et al., 2014; Güçlü and Gerven, 2015). While most researchers who relate deep language models to the brain use off-the-shelf language models, in this thesis we fine-tune deep language models with the intention of facilitating better analysis of language processing in the brain. We hypothesize that:

- Fine-tuning a deep language model to predict brain activity will induce representations which are more similar to those in the brain than the representations in off-the-shelf deep language models.
- Multitask learning will constrain the representations learned during fine-tuning so that they generalize better to unseen data.
- Multitask learning can be used to interpret how the fine-tuned model makes predictions
- Multitask learning can be used to generate hypotheses about the latent cognitive processes which drive brain activity.

Chapter 4 aims to address the first question of whether fine-tuning a deep language model results in representations of sequences of words which are better suited for predicting brain activity than the off-the-shelf deep language model. At first glance, this question might seem trivial, but it is quite possible that the quality of representations learned by the off-the-shelf model trained on huge corpora saturate the prediction performance on brain activity data. We show that this is not the case, and that indeed fine-tuning benefits prediction. The chapter also validates that the modifications to the representations in the model which are induced by training on brain activity generalize across participants, and gives some evidence that the modifications generalize across modalities. These lines of evidence support the idea that the information the model learns to encode during fine-tuning is not just idiosyncratic to the prediction of one person's brain activity or to a particular recording modality, but rather that the information is relevant to the cognitive processes underlying these observations, making the fine-tuned model a valid tool for probing those processes.

In chapter 3 we demonstrate that the pressure on the representations arising from multitask learning can be beneficial, showing that using multiple event-related potentials as training targets, or combinations of event-related potentials and behavioral data, results in better predictions on unseen data than using only a single event-related potential as a training target.

Chapters 3 and 5 also address the question of using multitask learning as a mechanism for interpretation. Chapter 3 primarily considers task interference as a method for relating tasks, and discusses how the results produced by the task interference method relate to the literature on event-related potentials. Chapter 5 primarily considers how brain activity predictions relate to natural language processing task predictions in terms of how the model combines latent representations together to make the two kinds of predictions.

Scientific questions. While the emphasis of this thesis is on the methodology, and while the methodology is most useful for exploratory analysis, we nonetheless set out to generate hypotheses about language induced brain activity recordings and the underlying processes that generate them. To that end we wish to generate hypotheses about the relationship between:

- Different kinds of neural responses, such as between the LAN and the P600 (two event-related potentials, i.e. signals derived from electroencephalography recordings), or between the LAN and

fMRI activity.

- A neural response and a behavior, such as between the LAN and the amount of time fixating on a word.
- A stimulus and an elicited behavior, such as whether when a word acts as the patient in a sentence, a person is more likely to look backwards in the text.
- A stimulus and evoked neural activity, such as sentence-length and the fMRI response.

Chapter 3 addresses several of these questions based on task interference during learning. In that chapter we find some relationships between neural responses which are expected from the literature (that the ELAN and LAN ERP components are related to the P600), and we speculate about some novel relationships that we find, as well as about the relative isolation of the N400 component from the other components. Chapter 3 also discusses the relationships between behavioral data and ERP components, in particular speculating about the relationship between latent look-forward/look-back operations, the several ERP components, and eye-tracking data, along with a different relationship between ERP components and self-paced reading time as a more generic marker of difficulty.

Chapters 4 and 5 focus more on the latter two questions, namely the relationships between a stimulus and an evoked neural response or behavior. In chapter 4, we compare examples where prediction accuracy changes the most during fine-tuning to examples where prediction accuracy doesn't change as much by examining the distributions of hand-coded labels on those sets of examples (Wehbe, Murphy, et al., 2014). From this comparison, we generate hypotheses about what kinds of features in the stimuli might be encoded differently in the brain than in an off-the-shelf version of BERT (a deep language model) (Devlin et al., 2018). In particular, we speculate that sequences of words pertaining to movement might be represented differently in the brain than in BERT. In chapter 5, we use task similarity to compare predictions of ERP components, eye-tracking measures, self-paced reading time, and fMRI data to natural language processing predictions. We find that ERP components are related to patient-like properties of sentences, while eye-tracking measures and self-paced reading time appear to be more influenced by modifiers in sentences. Both the patient-like properties and the presence of modifiers might be related to cognition-relevant data by the increased demands of meaning integration in more semantically complicated sentences. fMRI data also appears to be related to agent-like and patient-like properties of sentences along with sentence-length, but the relationship is relatively weak, suggesting that most of the information the model uses to predict fMRI is orthogonal to the set of NLP tasks we included. The gap between fMRI voxel-to-voxel task similarity and fMRI to NLP task similarity suggests an opportunity to continually update the set of tasks included in the method to explain more and more of the brain activity data in terms of interpretable tasks.

1.4 Organization of the thesis

The remainder of the thesis is organized as follows. Chapter 2 gives background on several topics which are relevant to this thesis. It discusses current theories of language processing in the brain, introduces some psycholinguistic results which suggest what kind of information about language processing might be latent in behavioral data, introduces event-related potentials and theories of their significance, and briefly introduces magnetoencephalography (MEG) and functional magnetic resonance imaging (fMRI), two technologies that provide the data about neural activity from which our brain activity tasks are derived. Chapter 3 provides a first demonstration of using multitask learning to analyze language processing in the brain. In it, we analyze event-related potentials (ERP components) together with eye-tracking data and self-paced reading times, where we use a single model to predict all of them. Chapter 4 takes a deeper look at fine-tuning a deep language model to predict brain activity. We provide evidence that fine-tuning improves

prediction compared to an off-the-shelf model, and that this improvement transfers across experiment participants. We also find weak evidence for transfer from prediction of MEG to prediction of fMRI. Together this evidence supports the hypothesis that during fine-tuning a model learns to encode information relevant to prediction of the cognitive processes underlying language, not just idiosyncratic relationships between text and a single person's brain activity, suggesting that the model can be used to analyze cognitive processes. Chapter 5 returns to the multitask framework, but uses model parameter similarity to compare different tasks to each other. That chapter better demonstrates the potential of using multitask learning, showing how task similarities can be used to better understand the mechanisms that a model uses to make its predictions and revealing interesting similarities between neural activity task predictions and natural language processing task predictions. Finally, we recapitulate what we have learned and conclude in chapter 6.

1.5 Summary of Contributions

- **Chapter 3:** We pioneer fine-tuning of deep neural networks pretrained on a language modeling task to predict event-related potential components derived from electroencephalography recordings of brain activity.
- **Chapter 3:** We combine behavioral data prediction and neural activity prediction together in a single model — a simple approach to leveraging heterogeneous cognition datasets.
- **Chapter 3:** We use constructive task-interference to generate hypotheses about how event-related potentials relate to each other and to behavioral data — a first approach to using multitask learning to explore cognition.
- **Chapter 3:** We show empirically that multitask learning reduces generalization error for event-related potential data. This holds for both training with multiple event-related potential components compared to training with a single event-related potential component, and for training with both behavioral and event-related potential data compared to event-related potential data alone.
- **Chapter 4:** We pioneer fine-tuning of deep neural networks pretrained on a language modeling task to predict functional magnetic resonance imaging (fMRI) recordings of brain activity.
- **Chapter 4:** We show empirically that fine-tuning outperforms regression from a pretrained deep model that is not specialized for brain activity prediction, suggesting that the model encodes information pertinent to brain activity into the parameters of the deep layers of the model — in other words that the representations are biased for brain activity prediction.
- **Chapter 4:** We compare a model fine-tuned to predict a single participant's brain activity to a model which has not been fine-tuned, finding that using regression from the former is better at predicting novel participant's brain activity than using regression from the latter, which suggests that the bias in the representations induced by fine-tuning is not idiosyncratic to a single participant's brain activity and therefore may be encoding something about latent cognitive processes.
- **Chapter 4:** We compare a model which has been fine-tuned to predict magnetoencephalography (MEG) recordings of brain activity to a model which has not been fine-tuned, finding that regression from the former is sometimes better and sometimes worse at predicting brain activity than using regression from the latter. Qualitatively, the data suggest that the bias in the representations induced by fine-tuning to predict MEG data can sometimes transfer to prediction of fMRI data, again suggesting that the model may encode something about latent cognitive processes.
- **Chapter 4:** We use language feature distributions to interpet how the representations in two deep

language models differ, and apply this method to explore how fMRI data biases model representations.

- **Chapter 5:** We pioneer jointly fine-tuning a model on behavioral, neural, and natural language processing data to bring together heterogeneous information across disciplines which is relevant to studying the cognitive processes underpinning language understanding in people.
- **Chapter 5:** We use the parameters of a model to assess the similarity between tasks in a multitask learning framework, and develop the covariance-informed cosine similarity to measure the similarity of model parameters.
- **Chapter 5:** We modify the MultiDDS task sampling algorithm (Wang, Tsvetkov, and Neubig, 2020) to scale to a large number of tasks and to express preferences for particular tasks.
- **Chapter 5:** We analyze the mechanisms a model uses to make predictions for natural language processing tasks using our task similarity method. We hypothesize that the model encodes information about the arguments in a proposition into the embedding of a verb and that in some cases the model uses competition between those arguments via the verb embedding to make a prediction. We further find that sequence-level embeddings contain detailed information about individual tokens in a sequence, and we elucidate the relationships between natural language processing tasks.
- **Chapter 5:** We apply our method to analyze event-related potential prediction, and find that those tasks are most similar to proto-patient tasks, suggesting that ERP variance is in part explained by the existence of patient-like concepts in propositions, possibly as an indicator of semantic complexity.
- **Chapter 5:** We apply our method to analyze eye-tracking and self-paced reading time data, and find that those tasks are influenced by word-length along with the presence of modifiers like adjectives and adverbs, again possibly indicating semantic complexity. We also hypothesize that self-paced reading time is influenced by the meaning integration induced when a word or phrase is modified by an adjective or adverb, and when punctuation delineates the end of a phrase.
- **Chapter 5:** We apply our method to analyze fMRI data, and find that fMRI prediction is influenced by proto-agent and proto-patient properties of a sentence, along with the sentence-length, but is somewhat orthogonal to the tasks we include in our model. Investigation of voxel-to-voxel task similarities reveals that there is a large gap between the information that is relevant to fMRI prediction and the information which is relevant to NLP task prediction, suggesting opportunity to continually refine the set of tasks included in the model to explain more and more of the fMRI activity.

1.6 Conclusion

In this thesis we examine how we can leverage deep learning to better understand language processing in the brain. By using multitask learning, we can naturally combine together heterogeneous datasets from across multiple disciplines (e.g. psycholinguistics, neuroscience, and natural language processing) to get different kinds of evidence about the cognitive processes involved in language processing. Furthermore, including multiple tasks in our models enables us to compare tasks to one another, and by so doing we get insight into the mechanisms the models use to make their predictions. Ultimately this insight into model mechanisms enables insight into the representations used by people during language processing, and in this thesis we apply the methods we develop to generate some hypotheses about what information is in those representations. We believe this work offers a novel and powerful analysis method for understanding how people process language.

Chapter 2

Background on Language Processing in the Brain and How It's Probed

Philosophers, linguists, psycholinguists, neuroscientists, and computer scientists have long studied the processes involved in language comprehension and production. The many approaches to the study of language include theories of generative grammars, information theory, systematic studies of deficits resulting from brain lesions, reaction time studies, and theories of conversational implicature. In short, researchers have taken strikingly different approaches to understanding language, many of which are complementary to each other. In this thesis, we consider several of the approaches together as sources of information about language which can be encoded into the parameters of a deep language model through multitask learning. Below, we discuss theories of language in the brain, focusing primarily on theories developed in neuroscience by observing deficits in people with brain lesions and running functional magnetic resonance imaging (fMRI) studies. We also briefly sketch how some of the other approaches to understanding language which are most relevant to this thesis can contribute to the understanding of language, and how our work fits into this context.

2.1 Language in the brain

This section, heavily based on Kemmerer (2014), aims to very briefly highlight some of the salient ideas from neuroscience that are most relevant to the interpretation of results in this thesis about how language is processed in the brain. There is a vast body of work studying language in the brain, but we mainly wish to emphasize a few points. First, while the network of brain regions involved in language comprehension is somewhat agreed upon, each region of the brain involved in that network appears to subserve multiple different functions and these are difficult to tease apart. Second, in general (but not always), regions that are more posterior in the brain are involved in low level and subconscious operations, such as processing phonemes or features of letters, and regions that are more anterior in the brain are involved in more effortful and attentive processes of integration, such as assembling the meanings of individual words into the larger meaning of a discourse. A potential exception to the rule that posterior regions are lower level is the involvement of temperoparietal cortex in the representation of situations and events. Third, certain aspects of meaning seem to be particularly relevant to the brain, and these aspects can inform our interpretation of prediction results.

2.1.1 Speech comprehension

The dual stream model. According to the dual stream model of speech comprehension (Hickok and Poeppel, 2007), input arrives in the cortex via primary auditory cortex in the dorsal posterior portion of superior temporal gyrus (pSTG) from subcortical regions which process sound waves. From there, two streams of processing occur. The dorsal stream, connecting pSTG to inferior frontal gyrus (IFG), premotor, and parietal areas, is responsible for mapping phonemes onto motor representations of the articulations that would be used to reproduce those sounds. Meanwhile, the ventral stream, connecting pSTG to posterior mid-temporal gyrus (pMTG) and inferior temporal sulcus, and ultimately to their anterior counterparts, maps phonemes onto semantic representations. The anterior part of the temporal lobe (ATL) is thought to integrate the various aspects of semantic meaning into a coherent representation (see word meaning below). While the dorsal stream is left dominant, the ventral stream is bilateral with some differences in function — the right hemisphere appears to be more involved in the processing of non-linguistic prosody while the left hemisphere appears to be more involved in linguistic prosody and the processing of semantics, but neither prosody or semantics is completely lateralized.

Prososody. The processing of emotional/affective prosody seems to be right hemishpere dominant (Beaucousin et al., 2007; Witteman, Van Heuven, and Schiller, 2012), with right ATL constructing tonal contours that are processed by the amygdala and right ventral frontoparietal areas involved in simulating the feelings those contours might represent. Executive/attentive processes in orbitofrontal and inferior frontal areas are also involved bilaterally (Adolphs, Damasio, and Tranel, 2002). Linguistic prosody (examples of linguistic prosody include changes in stress which differentiate nouns from verbs as in ***re****cord* and *record* or emphasize a word in context, pauses which indicate the placement of punctuation, or changes in tone which differentiate statements from questions) also involves areas near the primary auditory cortex along with anterior areas of temporal lobe (bilaterally), inferior frontal areas (bilaterally), and right inferior precentral sulcus (Hesling et al., 2005). The superior temporal regions are also thought to be the generators of an event-related potential (event-related potentials are described in section 2.2.2) known as the Closure Positive Shift (CPS) which is elicited by boundaries in sentences and phrases both in heard speech and when punctuation is encountered by readers (Steinhauer and Friederici, 2001). The CPS suggests that phrase boundaries trigger events in the brain, and also has been shown to vary idiosyncratically with punctuation habits (Steinhauer and Friederici, 2001).

2.1.2 Reading

Reading begins bilaterally in the visual system, with low level visual features building up to concrete instantiations of letters in the ventral occipital region (V4 of the visual system), and becoming abstractly represented as symbols in the anterior ventral occipital cortex (V8 of the visual system), finally moving into a left-dominant processing stream which combines letters into pairs and sequences in the left occiptotemporal sulcus and the visual word form area (VWFA) in the lateral portion of the left ventral occipotemporal sulcus (Dehaene et al., 2005; Plaut and Behrmann, 2011). Letter sequences are believed to be mapped onto both phonemes and semantics through separate pathways which are not very well understood, with the mapping onto sounds involving superior temporal cortex as well as frontoparietal regions, and the mapping onto meaning involving more inferior regions of the temporal cortex as well as inferior frontal gyrus (see figure 2.2) (Dehaene, 2009). These networks loosely correspond to the dorsal and ventral processing streams in the dual stream model of speech comprehension.

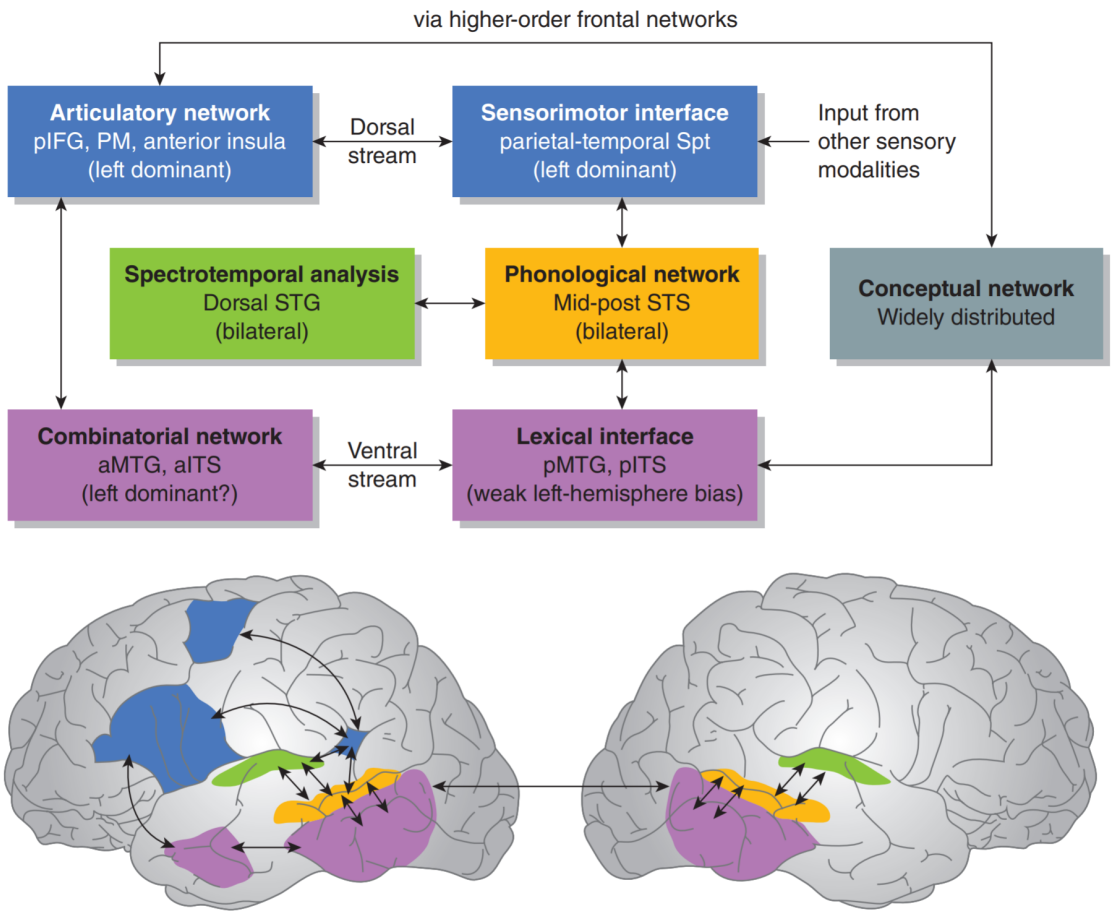


Figure 2.1: After low-level sound processing, phonemes are processed separately by two streams. The dorsal stream (in blue here) is largely left-lateralized and maps phonemes onto motor representations while the ventral stream (in pink here) is bilateral (with slightly different functions in each hemisphere), and maps phonemes onto semantic representations (Hickok and Poeppel, 2007).

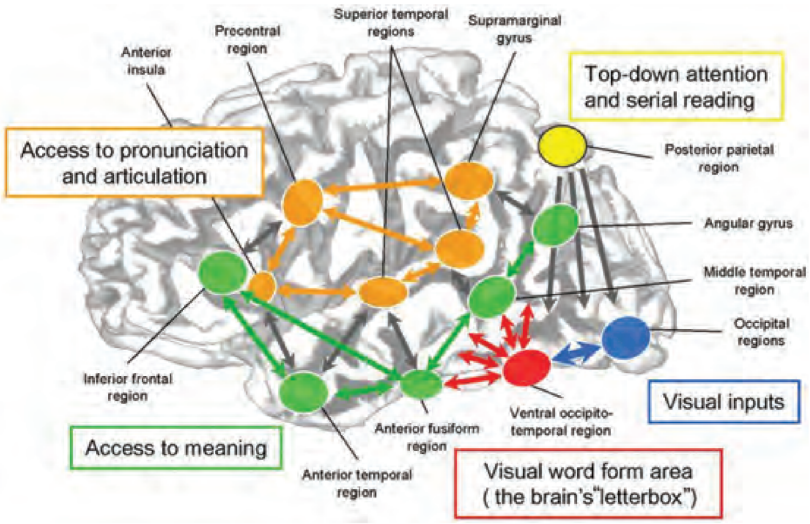


Figure 2.2: Once the visual features of letters and words are recognized, those visual forms are mapped onto both phonemes and semantics by two separate processing streams which are not well understood, but which loosely correspond to the dual stream model of speech comprehension (compare with figure 2.1) (Dehaene, 2009) reproduced from (Kemmerer, 2014).

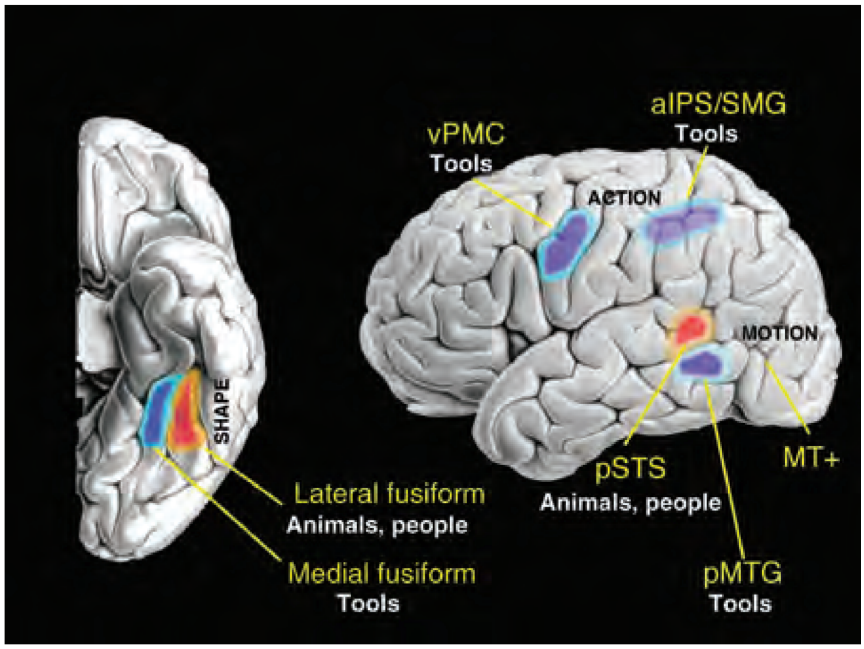


Figure 2.3: Certain areas of the brain are thought to encode features related to the shapes, motion features (how objects move), and motor features, (how objects can be manipulated) of object, and also appear to prefer stimuli from specific categories (Kemmerer, 2014).

2.1.3 Word meaning

Grounded cognition. The Grounded Cognition Model holds that concepts are represented in the brain not by symbolic feature representations like *yellow*, but by a recapitulation of the actual perceptual experience of the wavelength of light which corresponds to *yellow* falling on the retina. In general, the model suggests that visual features are supported by the visual cortex, auditory features by the auditory cortex, and so on. The concept *cup* is, in this model, essentially represented by the memories of interacting with cups. There is significant evidence to support this view, including domain-specific semantic impairments such as color agnosia (an inability to retrieve typical colors of objects) associated with lesions to specific brain areas (namely the left fusiform gyrus) (Simmons et al., 2007), fMRI studies of differential activation in tasks such as judging semantic similarity of objects from different categories (Goldberg, Perfetti, and Schneider, 2006), and prediction of fMRI images from words (Mitchell et al., 2008). Along these lines, concrete nouns seem to be particularly represented by perceptual features, including how those objects tend to move, and motor features (how an object is manipulated). Categories of nouns, such as tools and animals, also tend to activate slightly different areas within the appropriate perceptual areas (see figure 2.3). According to the hub and spoke model (Lambon Ralph et al., 2010), the anterior temporal lobe (ATL) integrates these different perceptual features into a higher level, more symbolic representation which can be used by other areas of the brain and which can still, for example, differentiate typical examples from atypical examples of an object category.

Verbs. In the grounded cognition model, verbs are also represented by memories. Motion features represented in posterolateral temporal cortex (PLTC) near the angular gyrus and extending into both parietal and superior temporal regions are particularly important, along with motor features in parietal and frontal regions associated with motor control. The brain is also sensitive to the level of transitivity of a verb, and the PLTC has been implicated in representing the causal relations in a transitive context, while Broca's area has been associated with processing the spatiotemporal relations between participants in transitive clauses (Ouden et al., 2009; Thompson et al., 2007).

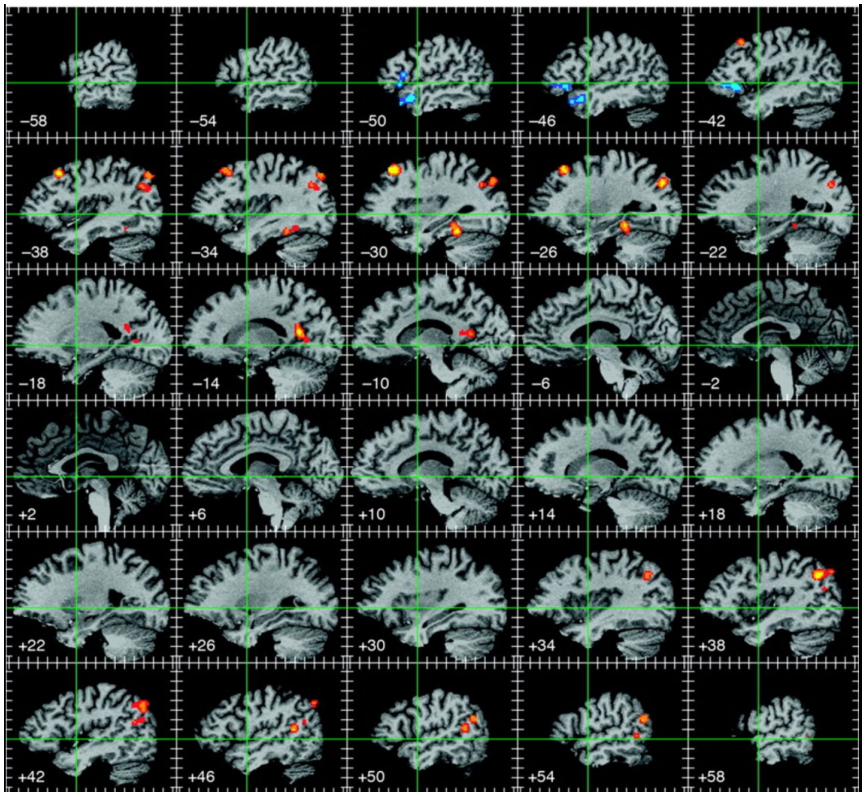


Figure 2.4: This contrast of abstract vs. concrete concepts, based on a meta-analysis shows activation associated with abstract concepts in cool colors and concrete activation in warm colors. Abstract concepts primarily activate the left IFG and and anterior superior temporal cortex (Binder et al., 2009).

Abstract concepts. The representation of abstract concepts is still controversial, but current findings suggest that they engage the cingulate gyrus and left fusifrom gyrus which subserve visual imagery, color, shape, spatial attention and navigation. Additionally the angular gyrus, associcated with integration, is activated relatively more by abstract than concrete concepts. Along with these regions, the inferior frontal gyrus (IFG) and anterior temporal lobe (ATL) are more active during the processing of abstract than concrete words, possibly in the role of applying a more concrete meaning to the abstract concept within the context in which the concept is invoked. (Binder et al., 2009; Hoffman, Jefferies, and Ralph, 2010; Wang, Conder, et al., 2010).

2.1.4 Integration

Morphosyntactic processing. Studies of morphosyntax typically use context to elicit an inflection as in "Every day I look at Susan. Just like every day, yesterday I _____ at Susan." (Ullman et al., 2005). Electrophysiological and fMRI studies both suggest that Broca's area (IFG) is involved in inflection of nouns and verbs in a sentential context during covert production of the inflected words, with some additional activation in the anterior insula and in the intraparietal sulcus (Sahin, Pinker, and Halgren, 2006; Sahin, Pinker, Cash, et al., 2009). The intraparietal sulcus is also implicated in the representation of number (Dehaene, 2009).

Sentences. Sentence processing involves all of the regions in the language network, and some investigation into the roles that each area plays in this processing has been undertaken. Posterior mid-temporal gyrus (pMTG) is thought to subserve lexical access by mapping sounds to semantics and to syntactic constraints (Hickok and Poeppel, 2007). It appears to be more active for category-ambiguous words (words that could be either nouns or verbs, for example) than for category-unambiguous words, which has been interpreted by some as multiple "lexical entries" being activated for ambiguous words which are later resolved by context (Snijders et al., 2008; Tyler et al., 2011). Anterior temporal lobe (ATL) may be involved in resolving these ambiguities, and has been proposed to integrate lexical entries into coherent representations and (alternatively) to group words into phrases through syntactic analysis (Dronkers et al., 2004; Brennan, Nir, et al., 2010). Meanwhile the posterior regions of the temporal lobe (pSTG, pSTS) extending into temperoparietal areas play a role in auditory short-term memory. They may also be involved in semantic (a.k.a thematic) role assignment by either directly inferring semantic roles, or alternatively through the recruitment of auditory short-term memory to help with semantic role assignment (Dronkers et al., 2004; Thothathiri, Kimberg, and Schwartz, 2012). Temperoparietal regions have additionally been implicated in arithmetic and spatial reasoning, and it has been suggested that semantic role assignment may use a general mechanism of spatiotemporal reasoning which this area implements (Baldo and Dronkers, 2007; Baldo, Bunge, et al., 2010; Chatterjee et al., 1995; Coslett, 1999). Finally, the inferior frontal gyrus (IFG) and ventrolateral prefrontal cortex (VLPFC), i.e. Broca's area and adjacent regions are very likely involved in sentence processing, but their role is unclear. One line of work suggests this area is primarily involved in syntactic extraction and analysis using a combination of semantics and syntax (see figure 2.5) (Bornkessel-Schlesewsky and Schlesewsky, 2012; Friederici, 2012). A second hypothesis is that this area engages in auditory rehearsal, utilizing auditory short-term memory to facilitate the analysis of complicated sentences (Rogalsky, Matchin, and Hickok, 2008). A third line of work suggests that the area is responsible for top-down, attentive control to resolve lower level ambiguities and that it does a final check of the syntactic and semantic analyses done primarily in other areas (Snijders et al., 2008; Tyler et al., 2011). Some evidence also suggests the role of this area is highly variable across different people (Prat and Just, 2011).

Discourse. At the discourse level, anterior regions in the right hemisphere appear to be crucial for producing stories with narrative coherence (see figure 2.6) (Ash et al., 2006). During story comprehension, compared to baseline conditions, several other areas are also more active. Orbitofrontal regions are active during stories (Troiani et al., 2008), and medial parietal cortex — which has been proposed as representing situational models (where events of stories are represented in the mind's eye) — shows increased activity both at the beginning of a story and at event boundaries (Speer, Zacks, and Reynolds, 2007). Along with medial parietal cortex, dorsomedial prefrontal cortex and temperoparietal junction are also more active during coherent than incoherent stories, and have been implicated in "theory-of-mind" tasks as well as maintenance of non-automatic cognitive processes related to coherence (Ferstl and Cramon, 2002; Amodio and Frith, 2016; Saxe and Kanwisher, 2003).

2.1.5 Summary of language in the brain

Language processing in the brain is still not very well understood, but there is some understanding of which regions are involved in the language network, and some inklings about the contributions that various regions and subnetworks might make to language processing. Figure 2.7 shows a visual summary of the contributions that have been attributed to various regions. Though this is by no means complete, it does provide a context in which to interpret the results in the thesis. In general there is a hierarchy of processing, where posterior temporal cortex is responsible for lower level processing, anterior temporal cortex and

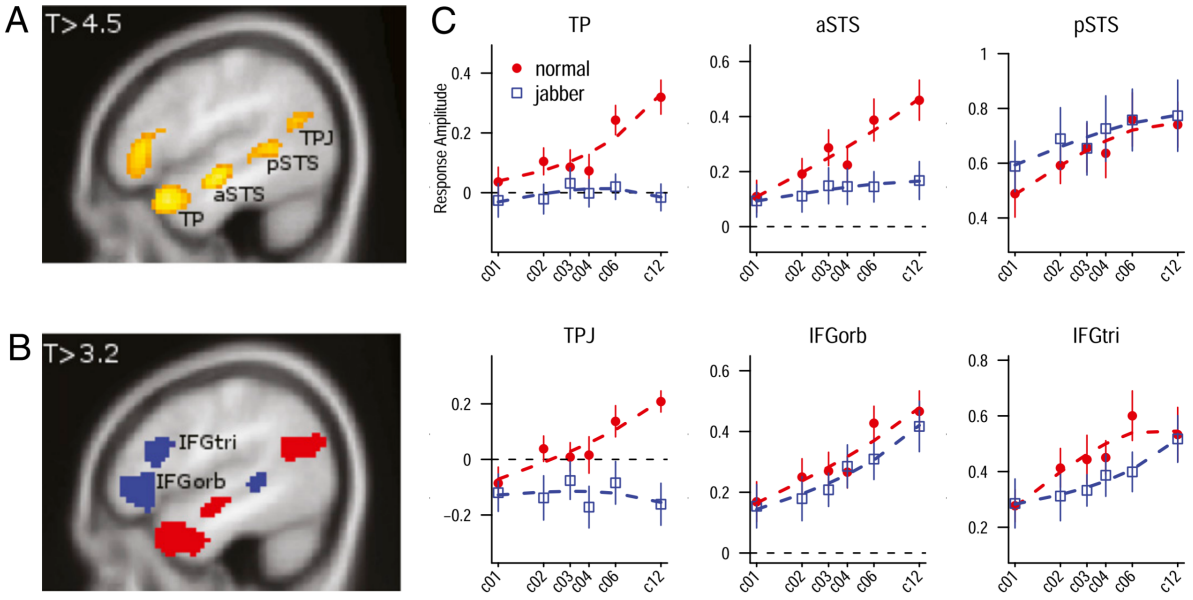


Figure 2.5: Pallier, Devauchelle, and Dehaene (2011) concatenate constituents of various sizes together to form pseudo-sentences, and observe that as the length of the constituents increases, the BOLD signal increases. In some regions (the temporal pole, temporal parietal junction, and anterior superior temporal sulcus — all shown in red), this sensitivity occurs only when actual words are presented, while in others (the pars orbitalis, pars triangularis, and anterior superior temporal sulcus — all shown in blue) this sensitivity also occurs when content words are replaced with pseudowords, suggesting that those regions may be processing hierarchical/syntactic information even in the absence of semantics (Pallier, Devauchelle, and Dehaene, 2011).

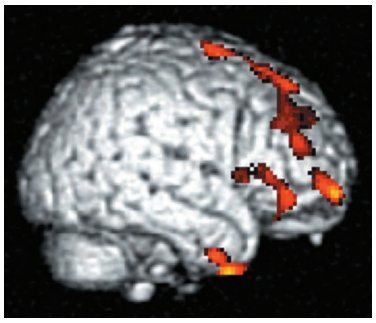


Figure 2.6: Patients with lesions in the highlighted areas had difficulty generating a coherent story from a set of picture prompts compared to patients with damage in different areas (Ash et al., 2006).

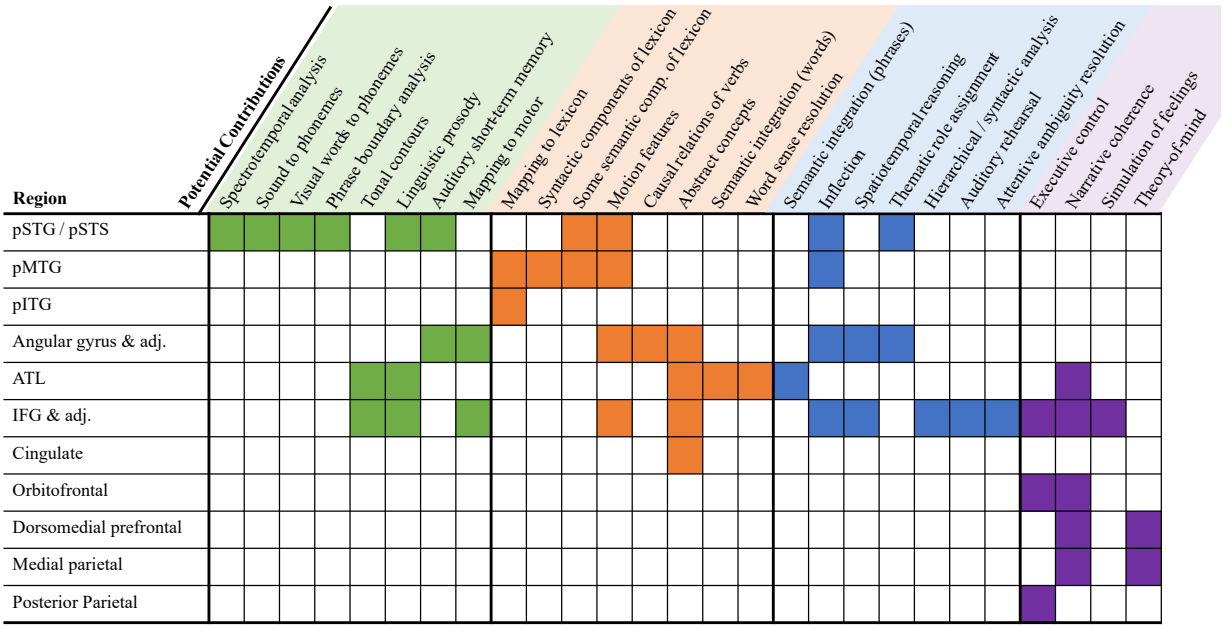


Figure 2.7: A summary of the contributions that various regions in the brain might make to language processing based on the current literature. pSTG and pSTS refer to the posterior superior temporal gyrus and sulcus respectively, pMTG refers to the posterior mid temporal gyrus, and pITG refers to the posterior inferior temporal gyrus. The row for angular gyrus and adjacent, includes area Spt (Sylvian parietal-temporal) of Hickok and Poeppel (2007), insula, and extends into adjacent parietal and temporal areas, including areas referred to as posterolateral temporal cortex (PLTC) and temporal parietal junction (TPJ). ATL denotes the anterior temporal lobe and includes anterior superior temporal gyrus (aSTG) as well as anterior mid temporal gyurs (aMTG). IFG and adjacent refers to the inferior frontal gyrus, and includes Broca's area, pars orbitalis, pars triangularis, pars opercularis, ventrolateral prefrontal cortex (VLPFC) as well as premotor areas. The colors roughly divide the contributions into auditory contributions, word meaning contributions, integration contributions, and discourse contributions. We can interpret results of the thesis in light of these theorized contributions.

inferior frontal gyrus are responsible for integration, and medial and lateral frontal and parietal regions work together on narrative coherence and theory-of-mind. Moving from the bottom to the top of this hierarchy also increases the time span of representations (from sounds and glyphs to words to sentences to stories) and increases in conscious awareness, where at the top of the hierarchy some of the operations require conscious attention and effort.

2.2 Probing language in the brain

Current understanding of how language is processed in the brain has largely come from the study of language deficits in people with lesions in various regions, but it has also been immensely helped by studies using electrophysiology and neuroimaging technologies as well as studies of behavior undertaken by psycholinguists. In this section, we describe some of these technologies, what they can contribute to our understanding of language processing, and their limitations. We use data from several of these technologies in the thesis.

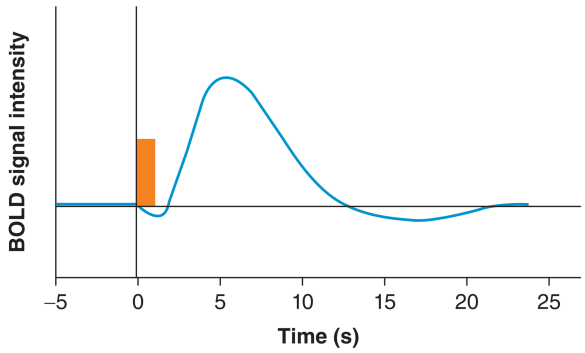


Figure 2.8: The hemodynamic response function. During the first 1 or 2 seconds after an area of the brain has become active and consumed the available oxygen, that area has a decreased blood oxygenation level dependent (BOLD) signal. Over the coarse of the next 5 seconds or so extra oxygen is sent to the area, and during that time the BOLD signal increases to its maximum. The signal returns to slightly below baseline over the next 5 seconds and then back to baseline. (Kemmerer, 2014)

2.2.1 Functional Magnetic Resonance Imaging (fMRI)

In magnetic resonance imaging (MRI), a magnetic field is applied to the brain which causes a parallel alignment of the magnetic fields of a small number of the hydrogen atoms in the water molecules that suffuse the brain tissue. Radio waves are then passed through the brain, which realigns the magnetic fields of those hydrogen atoms in a predictable way. When the radio waves are turned off, the magnetic fields of those hydrogen atoms return to a parallel alignment with the external magnetic field, and this "relaxation" is detected by sensors around the head. Because different tissues in the brain have different densities of hydrogen atoms, there is a contrast in this signal which can be used to reconstruct an image of the brain. This signal also turns out to be sensitive to the oxygenation level of the blood in the brain because the molecule in the blood which carries oxygen, hemoglobin, has a stronger magnetic field when it is not bound to oxygen than when it is bound to oxygen. When hemoglobin is not bound to oxygen, its magnetic field disrupts the MRI signal, and therefore deoxygenated blood reduces the MRI signal. This blood oxygenation level dependent (BOLD) response is the basis of most functional magnetic resonance imaging (fMRI). As neurons in the brain become active, they consume the available oxygen, which causes the BOLD signal to decrease in that area for 1 to 2 seconds. During the next 5 seconds, additional oxygen is sent to that area and the BOLD signal increases to its maximum. The next 5 seconds see a decrease in the BOLD signal back to slightly below the baseline, followed by a return to the baseline. This timecourse of the BOLD signal is called the hemodynamic response function (see figure 2.8). A major limitation of BOLD based fMRI is that it can only detect bloodflow and not actual neuronal activity. The signal is influenced by the locations of arteries, and the activity within a voxel only accounts for the blood flowing through arteries in that voxel, which represents a small portion of the voxel's volume. Additionally, the signal is distorted near air cavaties, and this can be especially problematic in areas of the brain near sinus cavaties, such as anterior temporal and orbitofrontal regions (Kemmerer, 2014).

2.2.2 Electroencephalography (EEG) and magnetoencephalography (MEG)

When a large number of neurons are active together, the electrical or magnetic fields generated by that activity can be measured from the outside of the head. Electroencephalography (EEG) uses electrodes placed on the scalp to measure the electrical fields, while magnetoencephalography (MEG) uses sensors (superconducting qauntum interference devices, or SQUIDs) to detect the tiny magnetic fields. Both EEG and MEG have very high temporal resolution and poor spatial resolution compared with fMRI, but both also more directly measure brain activity while fMRI only measures blood flow. The electrical fields measured by EEG are distorted by passing through the skull, cerebrospinal fluid, and scalp, so it is difficult to localize the area of the brain that generates a detected EEG signal. Magnetic fields do not suffer from

this distortion and have much higher spatial resolution than EEG, but because of the way the magnetic fields are aligned, MEG can only measure activity in the sulci of the brain and it is much less sensitive to activity in the gyri. Thus, fMRI, EEG, and MEG have complementary strengths and limitations. Often, EEG and MEG are used to measure stereotypical responses to stimuli known respectively as event-related potentials (ERPs) and event-related fields (ERFs). We discuss some of these below.

Event-related potentials and event-related fields

An event-related potential (ERP) or event-related field (ERF) is a stereotyped electrical or magnetic brain response that is synchronized with an external event. In this work, we consider six ERP responses that appear in our data and that have been associated in the literature with language processing: The N400, the P600, the LAN, the ELAN, the PNP, and the EPNP (see Figure 3.1). The N400, EPNP, and PNP responses are primarily considered markers for semantic processing, and the P600, ELAN, and LAN responses are primarily considered markers for syntactic processing. However, the precise nature of the interaction of the processes and the precise characterization of the ERP responses is not completely understood.

N400. The most well understood ERP is the N400 response. The accumulated evidence suggests that the N400 is a marker of "semantic effort" that is present by default and is attenuated by context (Kutas, Van Petten, and Kluender, 2006). The response is named for a negative deflection (relative to reference baselines) in the measured potential which occurs between 200ms and 600ms after stimulus onset and which attains its largest magnitude at around 400ms (Kutas and Federmeier, 2011). The response manifests in centro-parietal electrodes, and is primarily generated by the left and right temporal lobes, with fMRI especially implicating the superior temporal gyrus (STG) and middle temporal gyrus (MTG) (Kutas and Federmeier, 2011; Service et al., 2007; Van Petten and Luka, 2006). Perhaps most famously, the N400 has been correlated with the "cloze" probability — the proportion of individuals who would continue a sentence fragment with a given word (Kutas and Hillyard, 1984) — but more generally, the N400 is associated with semantic context at various levels of processing, and these levels interact. If words are presented in isolation, then the N400 is modulated by word frequency (Van Petten and Kutas, 1990). If a sentence context is available, then the sentence context supercedes the word frequency effect (Van Petten, 1993). For example, in the sentence *I like my coffee with cream and dog*, the word *dog* induces a large N400 effect despite being a common word. If a discourse-level context is available, the discourse-level effects supercede sentence-level effects (George and Mannes, 1994; Berkum, Hagoort, and Brown, 1999). For example the presentation of the sentence *Jane told the brother that he was exceptionally slow* in a discourse context where he had in fact been very quick elicits an N400 response (Berkum, Hagoort, and Brown, 1999). Real-world knowledge also plays a role in mediating the N400. Dutch people hearing that Dutch trains are white (they are actually yellow) exhibit a larger N400 than Dutch people hearing that Dutch trains are yellow (Hagoort, Hald, et al., 2004) and participants exhibit similar responses to knowledge acquired during an experiment (Fischler et al., 1985). The N400 has also been observed in non-verbal stimuli, again reducing when a stimulus is presented in context with semantically related stimuli (Kutas, Van Petten, and Kluender, 2006). Generally, the N400 is not affected by syntax or grammaticality (Allen, Badecker, and Osterhout, 2003).

P600. The P600 ERP is observed in centro-parietal locations between 500ms and 800ms post stimulus-onset, and occurs in response to syntactic violations of various kinds, including phrase-structure violations (*Bill admired Susan's of picture the park* (Kemmerer, 2014)) and pronoun case (*The plane took we to paradise and back* (Coulson, King, and Kutas, 1998)), as well as semantic-thematic violations in sentences like *For breakfast, the eggs would only EAT toast and jam* (Kutas, Van Petten, and Kluender, 2006;

Kuperberg, 2007; Kuperberg et al., 2003). It is thought to originate bilaterally in the temporal lobes, posterior to the N400 generator (but note that localization of the P600 uses functional characterization which is still controversial) (Service et al., 2007). Some researchers see the P600 as a generic (non-language specific) response to low-probability events — similar to the P300 family of responses (see (Folstein and Van Petten, 2011) for a review), while others see the P600 as a marker of syntactic integration difficulty (with variations in specific interpretations). For example, in Kuperberg (2007) the authors argue that the P600 is marker for a process, possibly under executive control, that is invoked or extended when there is conflict between two or more language processing streams that operate on lexicosemantic and morphosyntactic information respectively, a view also shared by Huettig (2015). This process may involve retrieving individual words from episodic memory (Van Petten and Luka, 2012).

ELAN. The early left anterior negativity (ELAN) is a controversial ERP that has been identified as functionally distinct from other ERPs and associated with a syntax-first model of language comprehension (Friederici, 2011; Hahne and Friederici, 1999). The response occurs about 100ms post stimulus-onset, and is thought by some to be a marker for word-category (e.g. noun or verb) expectation violations or sometimes phrase-structure violations in an early automatic process that builds a syntactic structure based on word-category information alone. Other researchers suggest that the ELAN is not a functionally distinct negativity, and is instead the early part of a sustained negativity that is frequently masked by the P600 (Kemmerer, 2014; Steinhauer and Drury, 2012). In this view, the sustained negativity is considered a marker for working memory demands in response to syntactic violations as well as violations involving temporal relationships (Kemmerer, 2014).

LAN. The left anterior negativity (LAN) is also controversial. It is not necessarily lateralized to the left hemisphere despite the name, and it occurs between 300ms and 500ms post stimulus-onset. The LAN has been characterized by some researchers as functionally distinct from the ELAN, with the ELAN marking word-category violations and the LAN marking morpho-syntactic agreement (for example subject-verb number agreement) violations (Friederici, 2002). Others have argued that the LAN occurs in response to word-category violations (Hagoort, Wassenaar, and Brown, 2003b). Still other researchers argue that the LAN is a marker for "look-back" or "look-forward" operations that resolve antecedents, find verb arguments, resolve syntactic violations, or perform similar operations; in short that it, like the ELAN, indexes working memory demands (Kutas, Van Petten, and Kluender, 2006).

EPNP/PNP. The post-N400-positivity (PNP) described by Van Petten and colleages is part of a biphasic response — it is characterized by an enhanced positivity that occurs just after an N400 (Van Petten and Luka, 2012). PNPs observed during sentences with incongruent words have mainly parietal scalp distributions, similar to the scalp distributions observed for P600 responses to semantic manipulations (as in *For breakfast, the eggs would only EAT toast and jam* described above), and it has been suggested that the P600 and parietal PNP are one and the same (Van Petten and Luka, 2012). In contrast, PNPs observed during sentences with congruent but low-probability words (RELAXATION in *On his vacation, he got some much needed REST / RELAXATION / SUN* (Van Petten and Luka, 2012)) have a more frontal distribution. The functional characterization of the PNP response remains speculative, but Thornhill and Van Petten (2012) and Van Petten and Luka (2012) suggest that it is a marker for a disconfirmed lexical prediction, as opposed to a disconfirmed concept. Kutas (1993) also suggests that this might be the manifestation of inhibition of a predicted word that was not realized. The early PNP (EPNP) is not functionally distinct from the PNP. It is a part of the PNP that is spatially more frontal, and therefore is less canceled by the N400. Consequently, this part of the PNP can be observed during the N400 (Thornhill and Van Petten, 2012).

2.2.3 Behavioral data

One way researchers have studied language in the brain is through observable behaviors such as reaction times, reading times, and eye movements while participants engage in language comprehension (Altmann, 2001; Traxler and Gernsbacher, 2011). These studies can help us understand a lot about the cognitive processes in which the brain is engaged, and here we briefly introduce a few examples of what we can learn from behavioral data.

Eye tracking. Eye-movements are considered to be a sign of overt visual attention, and therefore are a valuable indicator of what information is being processed at a given point in time (Rayner, 2009). The two major kinds of eye-movements that people make while reading are saccades and fixations (Rayner, 2009). Saccades are quick movements of the eyes from one fixation to another and fixations are pauses during which the eyes are looking at a single location. Regressions are saccades in the direction against the movement of reading back to earlier parts of the text. As reading gets more difficult (e.g. if low-frequency words are encountered or if syntax becomes complicated), then saccades become shorter, regressions become more frequent, and fixations get longer (Rayner, 1998). Several different measures of how much attention a reader places on a word or region of text have been developed and are found in our data. To study word-recognition processes, researchers typically use metrics that primarily measure early processing. Two such metrics are first-fixation duration and gaze duration. The first-fixation duration is the duration of the first fixation on a word and the gaze duration is the sum of the durations of all fixations on a word which occur prior to a fixation on any other word (to capture multiple fixations on the same word). For higher level processes and comprehension, the go-past time is a more popular metric. The go-past time is the sum of all durations on any word which precedes a target word starting from the first fixation on the target word and ending with the first fixation on any word which follows the target word. It is generally assumed that the go-past time captures the amount of time it takes a reader to integrate the meaning of a word into the ongoing context (Rayner and Pollatsek, 2006).

In addition to tracking eyes during reading, other types of eye-tracking studies can also give insight into language processing in the brain. Although not used in this thesis, one such paradigm is the visual world paradigm, in which a participant sees one or more images on a screen while listening to spoken language relating to that scene. The locations where the eyes fixate, and the timing of eye-movements relative to the spoken language can give insight into how cognitive processes unfold during speech comprehension (see figure 2.9).

An advantage of eye-tracking data compared to recordings of brain activity is that the raw eye-tracking data is relatively interpretable compared to fMRI or EEG data.

Reaction times and accuracy. Reaction times and accuracies can also provide a valuable window into the cognitive processes involved in understanding language. For example, several researchers have used a lexical decision task, in which the participant must decide whether a stimulus is a word or not, to study the relationship between phonology, orthography, and semantics in the brain (Gonnerman, Seidenberg, and Andersen, 2007; Rastle, Davis, and New, 2004; Feldman and Andjelković, 1992). Typically in these studies, a priming stimulus is used which overlaps with the target stimulus in the aspect of interest, e.g. phonology or semantics. If the participant is able to make a lexical decision more quickly and/or more accurately when the prime overlaps in this aspect than when it does not (i.e. the prime facilitates the decision), then researchers infer that the main task (lexical decision) is composed in part of the facilitating sub-task (e.g. recognizing phonemes). More relevant to this thesis is using reaction time as a signal of task difficulty. In studies utilizing self paced reading times, the experiment participant reads a text word-by-word or phrase-by-phrase and presses a button to advance to the next word or phrase. Much like the data from

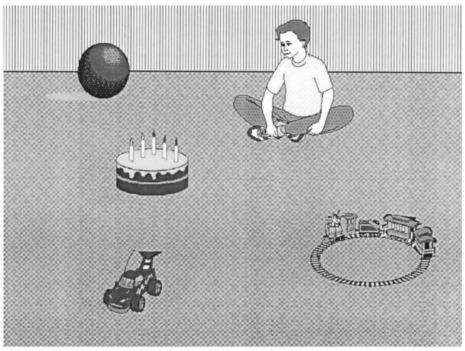


Figure 2.9: In the visual world paradigm, participants hear one of two sentences while viewing a scene. One of the two sentences has a verb which restricts the domain of the post-verbal noun. While viewing this example, participants hear either "The boy will eat the cake" or "The boy will move the cake" as their eye movements are tracked. In the *eat* condition, participants tend to fixate on the cake (the only object which in the scene which can be eaten) before the post-verbal noun is spoken. The timing suggests that the participants compute the semantics of the verb and the probability distribution over the likely post-verbal noun as soon as the verb is spoken (Altmann and Kamide, 1999).

eye-tracking, these self-paced reading times provide a signal of how difficult the word or phrase is to integrate into the larger text (Just and Carpenter, 1980; Just, Carpenter, and Woolley, 1982).

2.3 Computational modeling

Along with research that directly probes human behavior and processing in the brain, researchers have studied language by creating computational models of language processing. These computational models lie on a continuum from simulations — which are very much constrained in their structure or in their learning strategy by hypotheses about cognitive processes and biological considerations — to natural language processing (NLP) models which are only given a high-level task to perform and which have constraints only in-so-far as those constraints help to accomplish the task. In the former category a long tradition of connectionist models exist, perhaps most famously discussed by Rumelhart, McClelland, Group, et al. (1987), that attempt to show how the brain could accomplish certain tasks like converting a stream of letters into a stream of words (Elman, 1990), converting a stream of sounds into a stream of words (McClelland and Elman, 1986), or to predict an observed phenomenon, like the N400 (Laszlo and Plaut, 2012). Some models, like the EZ-reader model of Reichle, Rayner, and Pollatsek (2003) use explicit functions to make predictions. To the extent that they are hypothesis driven, simulations have the advantage of being very interpretable, and they allow researchers to better understand how a hypothesized cognitive process would work in practice. However, they are difficult to scale, and potentially overconstrained by the hypothesis space being explored by a researcher, which can prevent them from working well (for example the HMAX model for vision (Riesenhuber and Poggio, 1999) does not work nearly as well as a modern deep network architecture for computer vision). At the other end of the continuum, NLP models have recently shed more and more of their hypothesis driven constraints. There has been a movement towards end-to-end learning, which is completely driven by the statistics in the data. Models based on the transformer architecture (Vaswani et al., 2017), such as BERT (Devlin et al., 2018), and models based on the long short-term memory architecture (LSTM) (Hochreiter and Schmidhuber, 1997), such as ELMo (Peters et al., 2018), and which, critically, have been pretrained on very large corpora have been incredibly successful in solving natural language tasks and have touched off a new intense period of research which is quickly making further improvements. The extent to which the processing in these models is similar to human processing of language is not well understood, and indeed it is likely to be quite different from human processing, but nonetheless the models must be capturing information that is relevant to human language processing in order to perform the tasks on which they are trained.

2.4 Conclusion

In this thesis, we explore how language is processed in the brain using multitask learning and fine-tuning of deep language models. This chapter describes some of the literature about language processing in the brain in order to put our results (primarily from chapter 5) into context. Additionally, in order to improve the reader's understanding of what the data in our modeling represents, we have described some of the technologies that are used to procure that data. Finally, we discussed computational modeling since our work is most closely related to the tradition of understanding language through computation. While some computational modeling has attempted to learn about the brain by imposing constraints on the model, our work is closer to NLP modeling that focuses on performing the task at hand as well as possible by using optimization to implicitly learn a function from the input to the output. However, by making some of the outputs of our model be brain activity itself (or proxies thereof), we encourage our models to use representations which are more similar to representations used by the brain. A model which can successfully predict brain activity must use representations which are similar to the representations in the brain in the sense that those representations must capture the same information as the brain activity represents in order to predict it well. A similar argument applies to behavioral level data — by forcing the model to predict that data, we encourage the model to capture the same information as the latent cognitive processes that give rise to those observations. In the limit of having enough samples and enough different views of the underlying cognitive processes, we would expect that a model which can predict a large proportion of the variance of brain activity has a good model of the cognitive processes. Thus this thesis relates to prior computational modeling of language, but uses a somewhat different approach to discovering the cognitive processes which support language comprehension.

Chapter 3

Using Multitask Learning To Characterize Event-Related Potentials

This chapter is based on the work published as:
Dan Schwartz and Tom Mitchell. "Understanding language-elicited EEG data by predicting it from a fine-tuned language model". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (2019), pp. 43–57

In this chapter, we examine one approach to using a multitask framework for the analysis of language processing in the brain. Here, as each participant reads single sentences, that participant's brain activity is recorded with electroencephalography (EEG). Event-related potentials (ERPs) are derived from this EEG data, and each ERP component is treated as a prediction task in a multitask learning procedure. We show that we can fine-tune a deep language model to predict these components. Unlike previous approaches to ERP prediction, this approach can successfully predict (above chance levels) all of the ERP components. In addition to the ERP component prediction tasks, we also include eye-tracking measures and the pace of reading as prediction tasks. The relationship between ERP components, in the sense of which ERP component tasks benefit the learning of other ERP component tasks, can generate hypotheses about which ERP components are driven by similar underlying language processing in the brain. Similarly, when the prediction of behavioral data benefits the prediction of ERP components, those relationships are suggestive of the underlying processing which drives both measures. Thus, this multitask approach is both more successful in predicting ERP components than previous approaches and the relationships between tasks serve as a method of interpretation. Though this is only a step in the direction of understanding language processing in the brain, it demonstrates the potential of using multitask learning to that end.

3.1 Introduction

The cognitive processes involved in human language comprehension are complex and only partially identified. According to the dual-stream model of speech comprehension (Hickok and Poeppel, 2007), sound waves are first converted to phoneme-like features and further processed by a ventral stream that maps those features onto words and semantic structures, and a dorsal stream that (among other things) supports audio short-term memory. The mapping of words onto meaning is thought to be subserved by widely distributed regions of the brain that specialize in particular modalities — for example visual aspects of the word *banana* reside in the occipital lobe of the brain and are activated when the word *banana* is heard

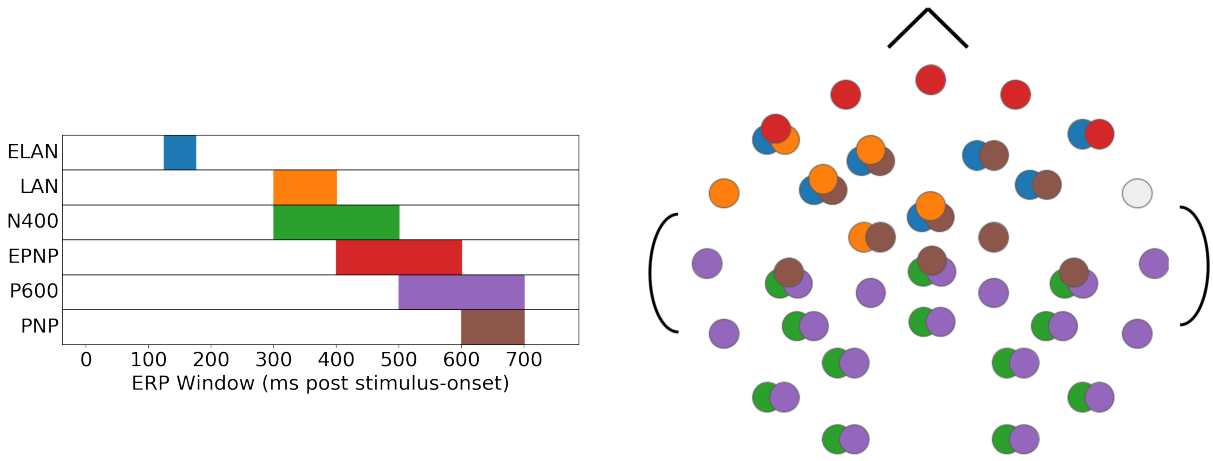


Figure 3.1: The electrodes from which each event-related potential was recorded in the data from Frank, Otten, et al. (2015) (after figure 3 in (Frank, Otten, et al., 2015)). The bottom portion of the figure shows a top-down schematic of the electrode locations with the nose facing towards the top of the page. Each ERP is the mean potential from all of the indicated electrodes during a specific time-window, creating a single scalar value per ERP per word. Overlapping circles indicate multiple ERPs recorded from the same electrode. The ELAN is measured from 125-175ms after stimulus onset, the LAN from 300-400ms, the N400 from 300ms-500ms, the EPNP from 400-600ms, the P600 from 500-700ms, and the PNP from 600-700ms.

(Kemmerer, 2014) — and the different representation modalities are thought to be integrated into a single coherent latent representation in the anterior temporal lobe (Lambon Ralph et al., 2010). While this part of meaning representation in human language comprehension is somewhat understood, much less is known about how the meanings of words are integrated together to form the meaning of sentences and discourses. One tool researchers use to study the integration of meaning across words is electroencephelography (EEG), which measures the electrical activity of large numbers of neurons acting in concert. EEG has the temporal resolution necessary to study the processes involved in meaning integration, and certain stereotyped electrical responses to word presentations, known as event-related potentials (ERPs), have been identified with some of the processes thought to contribute to comprehension.

In this work, we consider six ERP components that have been associated in the cognitive neuroscience and psycholinguistics literature with language processing and which we analyze in the data from Frank, Otten, et al. (2015) (see Figure 3.1 for spatial and temporal definitions of these ERP components). Three of these — the N400, EPNP, and PNP responses — are primarily considered markers for semantic processing, while the other three — the P600, ELAN, and LAN responses — are primarily considered markers for syntactic processing. However, the neat division of the ERP responses into either semantic or syntactic categories is controversial. The N400 response has been very well studied (for an overview see (Kutas and Federmeier, 2011)) and it is well established that it is associated with semantic complexity, but the features of language that trigger the other ERP responses we consider here are poorly understood. Here, we use a neural network pretrained as a language model to probe what features of language drive these ERP responses, and in turn to probe what features of language mediate the cognitive processes that underlie human language comprehension, and especially the integration of meaning across words.

3.2 Background

While a full discussion of each ERP component and the features of language thought to trigger each are beyond the scope of this chapter, please see chapter 2 for a longer discussion. Reviews are also available in e.g. Frank, Otten, et al. (2015), Kemmerer (2014), Kutas and Federmeier (2011), Kuperberg et al. (2003), and Van Petten and Luka (2012). Here, we introduce only some basic features of ERP components to help in the discussion later. ERP components are electrical potential responses measured with respect to a baseline that are triggered by an event (in our case the presentation of a new word to a participant in an experiment). The name of each ERP component reflects whether the potential is positive or negative relative to the baseline. The N400 is so-named because it is **N**egative relative to a baseline (the baseline is typically recorded just before a word is presented at an electrode that is not affected by the ERP response) and because it peaks in magnitude at about **400**ms after a word is presented to a participant in an experiment. The P600 is **P**ositive relative to a baseline and peaks around **600**ms after a word is presented to a participant (though its overall duration is much longer and less specific in time than the N400). The post-N400 positivity (PNP) is so-named because it is part of a biphasic response; it is a positivity that occurs after the negativity associated with the N400. The early post-N400 positivity (EPNP) is also part of a biphasic response, but the positivity has an eariler onset than the standard PNP. Finally, the LAN and ELAN are the left-anterior negativity and early left-anterior negativity respectively. These are named for their timing, spatial distribution on the scalp, and direction of difference from the baseline. It is important to note that ERP components can potentially cancel and mask each other, and that it is difficult to precisely localize the neural activity that causes the changes in electrical potential at the electrodes where those changes are measured.

3.3 Related Work

This work is most closely related to the paper from which we get the ERP data: Frank, Otten, et al. (2015). In that work, the authors relate the surprisal of a word, i.e. the (negative log) probability of the word appearing in its context, to each of the ERP signals we consider here. The authors do not directly train a model to predict ERPs. Instead, models of the probability distribution of each word in context are used to compute a surprisal for each word, which is input into a mixed effects regression along with word frequency, word length, word position in the sentence, and sentence position in the experiment. The effect of the surprisal is assessed using a likelihood-ratio test. In Hale et al. (2018), the authors take an approach similar to Frank, Otten, et al. (2015). The authors compare the explanatory power of surprisal (as computed by an LSTM or a Recurrent Neural Network Grammar (RNNG) language model) to a measure of syntactic complexity they call "distance" that counts the number of parser actions in the RNNG language model. The authors find that surprisal (as predicted by the RNNG) and distance are both significant factors in a mixed effects regression which predicts the P600, while the surprisal as computed by an LSTM is not. Unlike Frank, Otten, et al. (2015) and Hale et al. (2018), we do not use a linking function (e.g. surprisal) to relate a language model to ERPs. We thus lose the interpretability provided by the linking function, but we are able to predict a significant proportion of the variance for all of the ERP components, where prior work could not. We interpret our results through characterization of the ERPs in terms of how they relate to each other and to eye-tracking data rather than through a linking function. The authors in Wehbe, Vaswani, et al. (2014) also use a recurrent neural network to predict neural activity directly. In that work the authors predict magnetoencephalography (MEG) activity, a close cousin to EEG, recorded while participants read a chapter of *Harry Potter and the Sorcerer's Stone* (Rowling, 1999). Their approach to characterization of processing at each MEG sensor location is to determine whether it is best predicted by the context vector of

the recurrent network (prior to the current word being processed), the embedding of the current word, or the probability of the current word given the context. In future work we also intend to add these types of studies to the ERP predictions.

3.4 Method

Data. We use two sources of data for this analysis. The primary dataset we use is the ERP data collected and computed by Frank, Otten, et al. (2015), and we also use behavioral data (eye-tracking data and self-paced reading times) from Frank, Monsalve, et al. (2013) which were collected on the same set of 205 sentences. In brief, the sentences were selected from sources using British English with a criterion that they be understandable out of context. We use the ERP component values as computed by Frank, Otten, et al. (2015) which have been high-pass filtered at 0.5 Hz to reduce correlation between ERP components and modulus transformed (John and Draper, 1980) to make the distribution of component values more normal. We do not use the 100ms pre-trial baseline which is made available by Frank, Otten, et al. (2015) and which they use as a separate input to the mixed effects regression. For more information about the ERP datasets and data collection procedures we refer the reader to the original papers. For the behavioral data, we use self-paced reading times and four eye-tracking measures. Self-paced reading time is considered a signal of integration difficulty (i.e. as it becomes more difficult to integrate the meaning of the current word into the context, the amount of time a reader spends on the current word increases). The eye-tracking measures are intended to capture both early effects (effects modulated primarily by properties of the word independent of its context, such as word frequency and word length) and late effects (effects modulated by the context in which the word is found, i.e. comprehension difficulty) in word processing (Rayner and Pollatsek, 2006). In both cases, the eye-tracking measures provide a signal of overt visual attention, which is thought to strongly correlate with covert perceptual attention in normal reading (Rayner, 2009). We log-transform the self-paced reading time and the eye-tracking measures.

Model. To predict the ERP signals in the data, we start with a 3-layer bidirectional LSTM-based language model encoder using the architecture found in Merity, Keskar, and Socher (2017) and pretrained on the WikiText-103 dataset (Merity, Xiong, et al., 2016) (we use the pretrained model from Howard and Ruder (2018)). The pretraining objective is to minimize the negative log-likelihood of the next word for the forward LSTM and the previous word for the reverse LSTM. The word-embeddings (input embeddings) in the encoder have 400 components, the hidden layer outputs have 1150 components each, and the context-embeddings output from the encoder have 400 components. The forward-encoder and backward-encoder are independently fine-tuned on the baby version of the British National Corpus (Consortium, 2005) to help with prediction of British English (both the ERP data and eye-tracking data use British English). During task training the two encoders' output embeddings are concatenated together and fed into a causal-convolution layer which combines each pair of adjacent timepoints into a single pair-embedding with 10 components. The causal-convolution (i.e. convolution which is left padded) ensures that the pair-embeddings are aligned so that the prediction targets correspond to the later word in the pair. In other words the pair can be thought of as representing the 'current' and 'previous' words together. A ReLU is applied to the pair-embedding before it, along with the word length and the log probability of the word, is fed into a linear output layer to predict each ERP and behavioral measure (see Figure 3.2). The convolution and linear layers are initialized using the default PyTorch (Paszke et al., 2017) initialization, i.e. the initialization proposed in He, Zhang, et al. (2015). The encoder portion of the model includes dropout as applied in Merity, Keskar, and Socher (2017), but we use different dropout probabilities when we fit the neural and behavioral data (the dropout probability on the input embeddings was 0.05, 0.4 on the input to the LSTM, 0.4 on LSTM hidden layers,

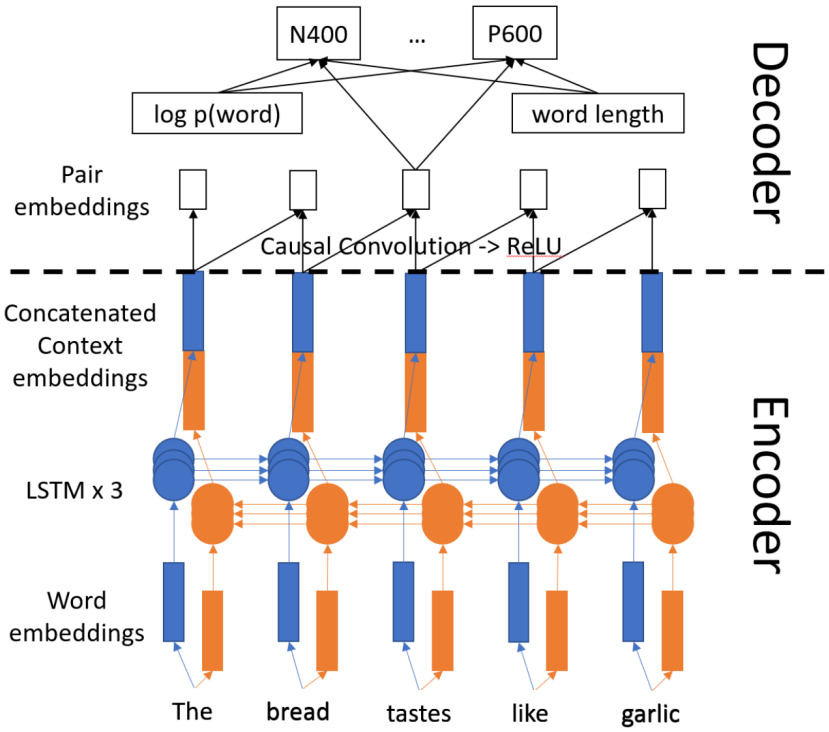


Figure 3.2: The model uses an encoder based on the architecture and regularization in Merity, Keskar, and Socher (2017) and pretrained by Howard and Ruder (2018). Within this architecture 2 independent 3-layer LSTM models encode a sentence. The context-embeddings output from each encoder are then concatenated together to give a single representation to each word in the sentence. These concatenated context-embeddings are fed into a causal-convolution, which learns a function to combine each pair of context-representations into a pair-embedding. A rectified linear unit (ReLU) non-linearity is applied to the pair-embedding, after which independent linear layers map the pair-embedding along with the log-probability of a word and the word-length to a prediction of each ERP or behavioral signal.

0.5 on the output of the LSTM, and 0.5 on the recurrent weights). We did not find dropout in the decoder to be helpful. We use the Adam optimizer (Kingma and Ba, 2014) with $\beta_1 = 0.95, \beta_2 = 0.999$ for training and we use mean squared error as the loss.

Procedure. We begin our training procedure by fine-tuning the forward- and backward-encoders independently on the baby version of the British National Corpus (Consortium, 2005). This corpus has British English that may help in modeling the University College London corpus, while not overlapping with it.

After the model fine-tuning, we estimate how well the model predicts each of the ERP signals and eye-tracking measures by training the model 100 times with different train/test splits and decoder parameter initializations. We use 10% of the data for testing and the remainder for training. The sentences in the ERP data are split at random. After we split the data, we compute the mean and standard deviation of each ERP signal (and each eye-tracking measure and the self-paced reading time) within participant on the training data. We use these values to standardize the training data within participant, and then average the data from all of the participants together. After we average, we again compute the mean and standard deviation to standardize the average. We follow a similar procedure for the test data, but we use the mean and standard

deviation from the training data when standardizing. Note that we use the log of the behavior measures, and the log is taken before the data-standardization.

In the loss function (and when we evaluate model performance) we only consider content words. We mark as a content word any word that is an adjective, adverb, auxiliary verb, noun, pronoun, proper noun, or verb (including to-be verbs). All other words are considered function words.

During the first 20 epochs of training, only the parameters of the decoder are modified. Following this, we train the model for an additional 15 epochs during which the parameters of the decoder and the final layer of the encoder (the final LSTM layer in both the forward and backward encoder) can be modified. We also experimented with additional training epochs and allowing all parameters of the model to be modified, but we found that this caused overfitting.

Comparing models trained with different loss functions. To better understand the relationship between ERP signals, and between ERP signals and behavioral data, we train the model with different loss functions that include mean squared error terms corresponding to various combinations of the ERP signals and behavioral data. For example, one of the training variations includes a mean squared error term for the P600 and a mean squared error term for the N400 in the loss, but does not use the other signals during training. In this variation, for a mini-batch of size B, where example b has T_b content tokens and the superscripts p and a denote the predicted and actual values for a measure respectively, the loss function can be written as:

$$\frac{1}{\sum_{b=1}^{B} T_b} \sum_{b=1}^{B} \sum_{t=1}^{T_b} (\text{P600}_{b,t}^{p} - \text{P600}_{b,t}^{a})^2 + (\text{N400}_{b,t}^{p} - \text{N400}_{b,t}^{a})^2$$

For each of the training variations, we repeat the training procedure described above (but fine-tuning the language model on the British National Corpus is done only once). We use a consistent train/test split procedure, such that the split for the ith run of the 100 runs is the same across all training variations, but the split changes between run i and run j. This enables us to use paired statistical testing when we test for significance.

We test for whether the proportion of variance explained (computed as $1 - \frac{\text{MSE}}{\text{variance}}$ on the validation set) on each ERP and behavioral measure is significantly different from 0 using the single sample t-test controlled for false discovery rate using the Benjamini-Hochberg-Yekutieli procedure (Benjamini and Yekutieli, 2001) with a false discovery rate of 0.01.

To test whether the proportion of variance explained is different between different training variations (for example training with just the N400 signal included in the loss vs. training with both the N400 and the LAN included in the loss), we use a paired t-test. We then adjust for the false discovery rate again with a rate of 0.01.

3.5 Results

All ERP components are predictable. In the original study on this dataset, the investigators found that when surprisal was used as a linking function between the language model and the mixed effects regression, the only ERP for which the surprisal showed a significant effect in the regression was the N400 (Frank, Otten, et al., 2015). In contrast, we find that when we directly predict the ERP signals we are able to predict a significant proportion of the variance for all of them (see Table 3.1).

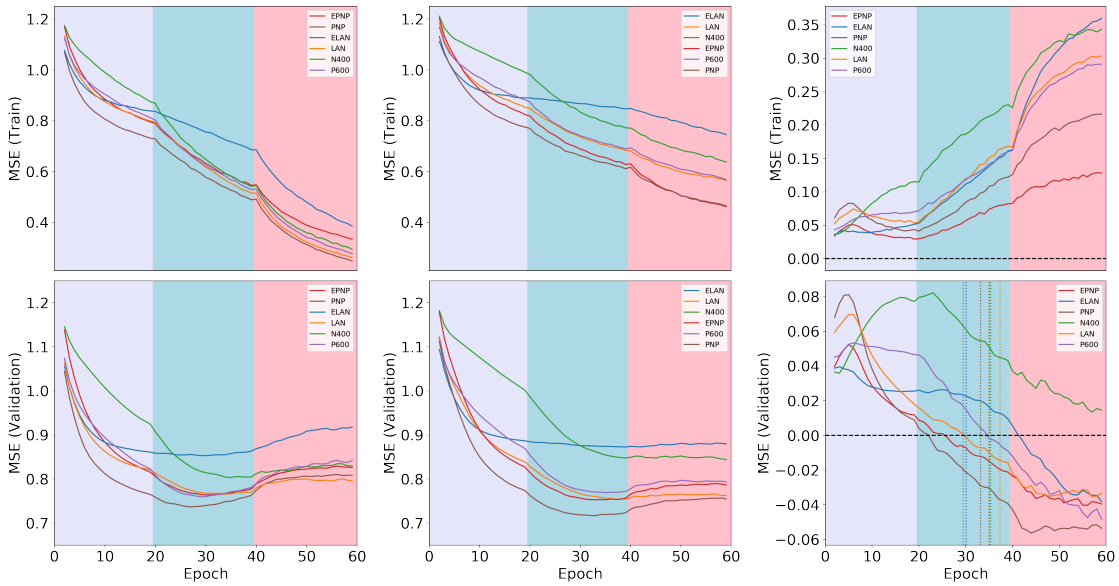
Joint training benefits ERP component prediction. To explore the relationship between ERP components, we train $63 = \binom{6}{1} + \binom{6}{2} + \cdots + \binom{6}{6}$ different models using all of the possible combinations of

Target	Additional	POVE	Target	Additional	POVE	Target	Additional	POVE
ELAN		0.20	LAN		0.30	N400		0.26
ELAN	+ EPNP	0.22	LAN	+ EPNP	0.31			
ELAN	+ N400	0.22	LAN	+ PNP	0.32			
ELAN	+ PNP	0.22	LAN	+ P600	0.32			
ELAN	+ P600	0.22	LAN	+ PNP, N400	0.33			
EPNP		0.34	P600		0.27	PNP		0.33
EPNP	+ LAN	0.35	P600	+ EPNP	0.30	PNP	+ LAN	0.36
EPNP	+ GROUP A	0.36	P600	+ LAN	0.30	PNP	+ GROUP B	0.36

Table 3.1: Proportion of variance explained (POVE) for each of the ERP components (mean of 100 training runs). The second column in each cell shows which ERP components in addition to the target ERP component were included in training. All combinations of training signals were explored. Shown is the best combination for each ERP target as well as every combination which is (i) significantly different from training on the target component alone, (ii) not significantly different from the best training combination, and (iii) uses no more than the number of signals used by the best combination. The N400 is predicted best when only the N400 signal is included in training. All values are significantly different from 0. GROUP A refers to (PNP, ELAN, LAN, P600) and GROUP B refers to (EPNP, ELAN, LAN, P600).

which of the six ERP signals are included in the loss function during training. For each of the six ERP components, we look for the best performing models (see Table 3.1). The N400 is best predicted when the model is trained on that component independently, but every other ERP component prediction can be improved by including a second ERP component in the training. Thus multitask learning has a clear benefit when applied to the ERP data and some information is shared between ERP component predictions via the model parameters. We also note that it is not the case that training with more ERP components is always better, or that the signals which are most correlated benefit each other most (see Appendix A). The relationship between components clearly impacts whether the prediction of one ERP component benefits from the inclusion of others in model training. The results suggest that 8 pairs of ERP signals are related to each other: the LAN is paired with the P600, EPNP, and PNP, the ELAN with the N400, EPNP, PNP, and P600, and the EPNP is paired with the P600. We discuss these relationships in section 3.6.

In an additional analysis, we modified our training procedure slightly to probe how jointly training on multiple ERP components compares to training individually on each ERP component. In this analysis we compare only training on each ERP component individually to training on all six ERP components together. We also train for a total of 60 epochs (rather than the 35 epochs used elsewhere). During the first 20 epochs we allow only the parameters of the decoder to be modified. During the next 20 epochs, we allow the parameters of the decoder and the final layer of the encoder (i.e. the final recurrent layer) to be modified. During the last 20 epochs, we allow all of the parameters of the model to be modified. The mean squared error for each of the ERP components from this analysis is shown for each epoch in Figure 3.3. From the loss curves, we make a few observations. First, we see inflection points at epochs 20 and 40, when we allow more parameters of the model to be modified. The first inflection point indicates that allowing the recurrent layer to be modified benefits the prediction, while the second inflection point shows that overfitting becomes more severe if we allow all parameters of the model to be modified. We also see from these curves that part of the benefit of joint training is that it helps reduce overfitting – we see less of a climb in the validation loss after the minimum point in the joint training. Beyond this reduction in overfitting severity, we note that for some of the ERP components (the LAN, EPNP and PNP components)



(a) Independently trained (b) Jointly trained (c) Joint - Indep.

Figure 3.3: The mean squared error (MSE) for prediction of each of the ERP signals during each epoch of training (mean of 100 training runs). The first 2 epochs have been omitted for clarity. During the first 20 epochs (lavender background), only the decoder parameters are modified. During the next 20 epochs (light blue background), the parameters in the final layer of the encoder are also modified. During the last 20 epochs (pink background), all of the parameters are modified. Note that in this model architecture, information can be shared between ERP signals even when only the decoder is modified. The figure shows the MSE when separate models are trained for each ERP independently (a), the MSE when a single model is trained on all ERPs jointly (b), and the difference between these two scenarios (c). The top row in each column shows the MSE on the training data while the bottom row shows the MSE on the validation data. In the bottom row right, the dotted vertical lines indicate the epoch at which the minimum MSE is reached in the lower of the independent or joint training. The LAN, EPNP, and PNP all show modest benefits from joint training before overfitting sets in (the minimum value occurs in the joint training scenario), while all ERP signals other than the N400 show reduced overfitting in joint training.

joint training actually gives a better overall minimum in prediction error.

Behavioral data benefits the prediction of ERP components. We are also interested in whether behavioral data can be used to improve ERP prediction since it should signal both the amount of overt attention required at various points in a sentence as well as integration difficulty. To study this question, we again train models using different combinations of training signals that include or do not include the behavioral data predictions in the loss function (see Table 3.2). We see that self-paced reading time indeed can improve prediction of a target ERP component relative to training on the target ERP component alone by about the same amount as the best combination of ERP components for all but the N400. Eye-tracking data can also improve the prediction accuracy of the ELAN, P600, and PNP components.

Target	Additional	POVE	Target	Additional	POVE	Target	Additional	POVE
ELAN		0.20	LAN		0.30	N400		0.26
ELAN	+ ERP	**0.22**	LAN	+ ERP	**0.33**	N400	+ ERP	0.26
ELAN	+ READ	**0.22**	LAN	+ READ	**0.31**	N400	+ READ	0.27
ELAN	+ EYE	**0.22**	LAN	+ EYE	0.30	N400	+ EYE	0.25
EPNP		0.34	P600		0.27	PNP		0.33
EPNP	+ ERP	**0.36**	P600	+ ERP	**0.30**	PNP	+ ERP	**0.36**
EPNP	+ READ	**0.35**	P600	+ READ	**0.29**	PNP	+ READ	**0.34**
EPNP	+ EYE	0.34	P600	+ EYE	**0.29**	PNP	+ EYE	**0.34**

Table 3.2: Proportion of variance explained (POVE) for each of the ERP components (mean of 100 training runs). +ERP indicates the best combination of ERP training signals for the target ERP component, + READ indicates the inclusion of self-paced reading times, +EYE indicates the inclusion of eye-tracking data, and bold font indicates a significant difference from training on the target component alone.

Insensitivity to choice of architecture. One potential concern about our results is the degree to which the relationships we see between ERP components and between ERP components and behavioral data is an artifact of our rather arbitrary choice of network architecture. We partially address this by running the same analysis using (i) only the forward direction of the encoder, and (ii) only the word-embeddings (the input embeddings) and not the context-embeddings (the output embeddings) of the encoder. The proportion of variance explained for each ERP component is lower using these variants of the analysis than using the bidirectional variant (see Appendix A), but qualitatively the relationships are similar. We leave further analysis of the sensitivity of our qualitative results to choice of architecture for future work.

3.6 Discussion

In this work we find that all six of the ERP components from Frank, Otten, et al. (2015) can be predicted above chance by a model which has been pretrained using a language modeling objective and then directly trained to predict the components. This is in contrast to prior work which has successfully linked language models to the N400 (Frank, Otten, et al., 2015) and P600 (Hale et al., 2018) but not the other ERP components. We also note that contrary to Hale et al. (2018), we find that an LSTM does contain information that can be used to predict EEG data, and in particular that it can predict the P600. We speculate that the analysis used in Hale et al. (2018) did not find reliable effects because the language models were related to the EEG data through functions chosen a priori (the surprisal, and the 'distance' metric). These functions, though interpretable, might be interpretable at the cost of losing much of the information in the representations learned by the network.

In addition, we show through our multitask learning analysis that information is shared between ERP components, and between ERP components and behavioral data. Although these relationships must be viewed with caution until they can be verified across multiple datasets and with more variation in neural network architectures, here we consider some potential reasons for our findings. The broad point we wish to make is that by better understanding which ERP components share information with each other and with behavioral data through the type of analysis we present here (multitask learning) or other means, we can better understand what drives each ERP component and in turn the processes involved in human language comprehension.

Relationship between ERPs. Our findings that the LAN and P600 are related, and that the ELAN and P600 are related are expected from both a theoretical perspective and from previous work examining the interactions of ERP components (Gunter, Stowe, and Mulder, 1997; Hagoort, Wassenaar, and Brown, 2003a; Hahne and Friederici, 1999; Kutas, Van Petten, and Kluender, 2006; Palolahti et al., 2005). Since the ELAN and LAN have been theorized by some to mark word-category (i.e. part-of-speech) or morpho-syntactic (e.g. subject-verb number agreement) violations (Friederici, 2011; Hahne and Friederici, 2002; Hagoort, Wassenaar, and Brown, 2003b) and the P600 is considered a marker for syntactic effort (Coulson, King, and Kutas, 1998; Huettig, 2015; Kemmerer, 2014; Kuperberg, 2007; Kuperberg et al., 2003; Van Petten and Luka, 2012), these signals would naturally be related to each other.

The other relationships we find are more surprising. Some researchers have speculated that the LAN and ELAN are markers for working memory demands (King and Kutas, 1995; Kutas, Van Petten, and Kluender, 2006), and that indeed these might be part of sustained negativities that are frequently masked by the P600 (Kemmerer, 2014). If we take this view, then we would expect to find them in the presence of semantic and syntactic complexity, and this might explain why they seem to benefit from joint training with the other ERP component signals (and benefit prediction of other ERP signals with which they are trained). However, it is notable that predictions of the LAN and ELAN do not benefit each other in our analysis, and that the N400 (a marker for semantic complexity) is not benefited by the prediction of any other ERP component. This absence is by no means definitive, but it undermines the argument that all of these relationships can be explained by complexity and working memory demands alone.

The relative isolation of the N400 from other ERP components in our analysis is interesting. If the N400 is a marker for semantic memory retrieval (Kutas and Federmeier, 2011), then it might be expected to be somewhat isolated from the other components, which may involve syntactic processing or later integration effects.

Alternatively, the relationships we find in our analysis might be an artifact of the way the ERPs are operationalized in Frank, Otten, et al. (2015). Several of the pairings we find overlap spatially and are near to each other in time, so the ERP components might spill over into each other. Further work is required to disambiguate between these possibilities.

Relationship between behavioral data and ERPs. It is reassuring to see that jointly training models to predict behavioral data along with a target ERP component benefits the prediction of the ERP component compared to training on the target ERP component alone. The benefit to prediction in this case cannot be explained as an artifact of how the ERP components are operationalized in the datasetes we use for analysis.

Self-paced reading times widely benefit ERP prediction, while eye-tracking data seems to have more limited benefit to just the ELAN, LAN, and PNP ERP components. It's difficult to know why this might be the case, but perhaps it is not a coincidence that these three ERP components also show up frequently in the pairs of components that benefit from joint training. If indeed the PNP marks semantic role irregularities (Van Petten and Luka, 2012) and the ELAN and LAN mark working memory or look-forward or look-back operations (Kutas, Van Petten, and Kluender, 2006), then its possible that eye-movements might be more related to these types of operations than to general semantic and syntactic complexities marked by other ERP components. Self-paced reading might better capture these generic difficulties. This explanation is highly speculative, and further work is required to determine whether the relationships between the ERP components and behavioral data are consistent across datasets, and if so, what the explanation is for these relationships.

Choice of bidirectional architecture. We emphasize that the neural network architecture we chose for these analyses was motivated primarily by its success on downstream NLP tasks, public availability of pre-trained models and code, and prior work studying how best to fine-tune the model (Howard and Ruder, 2018; Merity, Keskar, and Socher, 2017). We do not claim that this architecture reflects human processing. We experimented with a forward-only model variant of our analysis, and found that the bidirectional model predicts brain activity better than the forward-only version (see Appendix A). Although the bidirectional model has access to 'future' language input, it does not have access to future brain-activity, so the bidirectional model is not 'cheating' when it makes predictions. There are at least three possible explanations for why the bidirectional model performs better than the forward-only model. First, it is possible that when a human reads a sentence, he or she predicts the upcoming language input. Under this hypothesis, a model with access to the future language input can do a better job of predicting the current brain activity because the future language is reflected in that brain activity. Second, it is possible that a bidirectional model is simply able to produce better embeddings for each word in the input because it has more context than a forward-only model. For example, the bidirectional model might be (implicitly) better at anaphora resolution given more context. Under this hypothesis, the additional context given to the model partially compensates for its relative deficit of real-world knowledge compared to a human. Where a human can in many cases solve the anaphora resolution problem by using background knowledge and does not need to see the future language input, a model benefits from additional context. Finally, in our setup, the bidirectional model has more parameters than the forward-only model, and the additional degrees of freedom might give the model an advantage in predicting brain activity. Exploration of why the bidirectional model is better than the forward-only model is an interesting question, but it is left to future work. Additionally, as we noted earlier, the qualitative results of our analysis (e.g. how ERP components relate to each other) should be viewed with caution until they are replicated across multiple choices of architecture.

3.7 Conclusion

We have shown that ERP components can be predicted from neural networks pretrained as language models and fine-tuned to directly predict those components. To the best of our knowledge, prior work has not successfully used statistical models to predict all of these components. Furthermore, we have shown that multitask learning benefits the prediction of ERP components and can suggest how components relate to each other. At present, these joint-training benefit relationships are only suggestive, but if these relationships ultimately lead to insights about what drives each ERP component, then the components become more useful tools for studying human language comprehension. By using multitask learning as a method of characterization, we have found some expected relationships (LAN+P600 and ELAN+P600) and several more surprising relationships. We believe that this is exactly the kind of finding that makes multitask learning an interesting exploratory technique in this area. Additionally, we have shown that information can be shared between heterogeneous types of data (eye-tracking, self-paced reading, and ERP components) in the domain of human language processing prediction, and in particular between behavioral and neural data. Given the small datasets associated with human language processing, using heterogeneous data is a potentially major advantage of a multitask approach. In future work, we will further explore what information is encoded into the model representations when neural and behavioral data are used to train neural networks, and how these representations differ from the representations in a model trained on language alone.

Chapter 4

Generalizability of Models Fine-Tuned On Brain Activity Recordings

This chapter is based on the work published as:
Dan Schwartz, Mariya Toneva, and Leila Wehbe. "Inducing brain-relevant bias in natural language processing models". In: *Advances in Neural Information Processing Systems*. 2019, pp. 14100–14110

This chapter investigates several questions which are pertinent to our fine-tuning/multitask approach to the analysis of language processing in the brain. First, we show that a fine-tuned deep language model is better at predicting functional magnetic resonance imaging (fMRI) data than a simple regression from a frozen deep language model. This establishes that during fine-tuning, the parameters of the model do indeed capture information that is helpful for prediction of brain activity. Second, we show that the changes to the model parameters from training on one participant generalize to the prediction of held-out participants' fMRI data. We take this as an encouraging sign that the model's representations are better capturing brain-relevant information in general, not just idiosyncrasies in one individual. Third, by first fine-tuning the model to predict magnetoencephalography (MEG) data and then training the model to predict fMRI data we find some evidence that the information which the model's parameters capture during fine-tuning is relevant to prediction of language processing as opposed to the prediction of a specific brain-activity recording modality. Finally, we employ a method for the interpretation of how the deep language model parameters change during fine-tuning. We examine the distribution of language features in the examples most changed by fine-tuning as compared to the examples which are least changed. This method can generate hypotheses about representations of language in the brain in the absence of the joint multi-task learning as in chapters 3 and 5.

4.1 Introduction

The recent successes of self-supervised natural language processing (NLP) models have inspired researchers who study how people process and understand language to look to these NLP models for rich representations of language meaning. In these works, researchers present language stimuli to participants (e.g. reading a chapter of a book word-by-word or listening to a story) while recording their brain activity with neuroimaging devices (fMRI, MEG, or EEG), and model the recorded brain activity using representations extracted from NLP models for the corresponding text. While this approach has opened exciting avenues in understanding the processing of longer word sequences and context, having NLP models that are specifically designed to capture the way the brain represents language meaning may lead to even more insight.

We posit that we can introduce a brain-relevant language bias in an NLP model by explicitly training the NLP model to predict language-induced brain recordings, and that the information the model learns to capture during this training will generalize across multiple participants and brain recording modalities.

In this study we posit that a pretrained language model — BERT by Devlin et al. (2018) — which is then fine-tuned to predict brain activity will modify its language representations to better encode the information that is relevant for the prediction of brain activity. We further propose fine-tuning simultaneously from multiple experiment participants and multiple brain activity recording modalities to bias towards representations that generalize across people and recording types. We suggest that this fine-tuning can leverage advances in the NLP community while also considering data from brain activity recordings, and thus can lead to advances in our understanding of language processing in the brain.

4.2 Related Work

The relationship between language-related brain activity and computational models of natural language (NLP models) has long been a topic of interest to researchers. Multiple researchers have used vector-space representations of words, sentences, and stories taken from off-the-shelf NLP models and investigated how these vectors correspond to fMRI or MEG recordings of brain activity (Mitchell et al., 2008; Murphy, Talukdar, and Mitchell, 2012; Wehbe, Vaswani, et al., 2014; Wehbe, Murphy, et al., 2014; Huth et al., 2016; Jain and Huth, 2018; Pereira et al., 2018). However, few examples of researchers using brain activity to modify language representations exist. Fyshe et al. (2014) builds a non-negative sparse embedding for individual words by constraining the embedding to also predict brain activity well, but most approaches combining NLP models and brain activity do not modify language embeddings to predict brain data.

Because fMRI and MEG/EEG have complementary strengths (high spatial resolution vs. high temporal resolution) there exists a lot of interest in devising learning algorithms that combine both types of data. One way that fMRI and MEG/EEG have been used together is by using fMRI for better source localization of the MEG/EEG signal (He, Sohrabpour, et al., 2018) (source localization refers to inferring the sources in the brain of the MEG/EEG recorded on the head). Palatucci (2011) uses CCA to map between MEG and fMRI recordings for the same word. Mapping the MEG data to the common space allows the authors to better decode the word identity than with MEG alone. Cichy, Pantazis, and Oliva (2016) propose a way of combining fMRI and MEG data of the same stimuli by computing stimuli similarity matrices for different fMRI regions and MEG time points and finding corresponding regions and time points. Fu et al. (2017) proposes a way to estimate a latent space that is high-dimensional both in time and space from simulated fMRI and MEG activity. However, effectively combining fMRI and MEG/EEG remains an open research problem.

4.3 Methods

4.3.1 MEG and fMRI data

In this analysis, we use magnetoencephalography (MEG) and functional magnetic resonance imaging (fMRI) data recorded from people as they read a chapter from *Harry Potter and the Sorcerer's Stone* Rowling, 1999. The MEG and fMRI experiments were shared respectively by the authors of Wehbe, Vaswani, et al. (2014) at our request and Wehbe, Murphy, et al. (2014) online[1]. In both experiments the chapter was presented one word at a time, with each word appearing on a screen for 0.5 seconds. The chapter included 5176 words.

[1] http://www.cs.cmu.edu/~fmri/plosone/

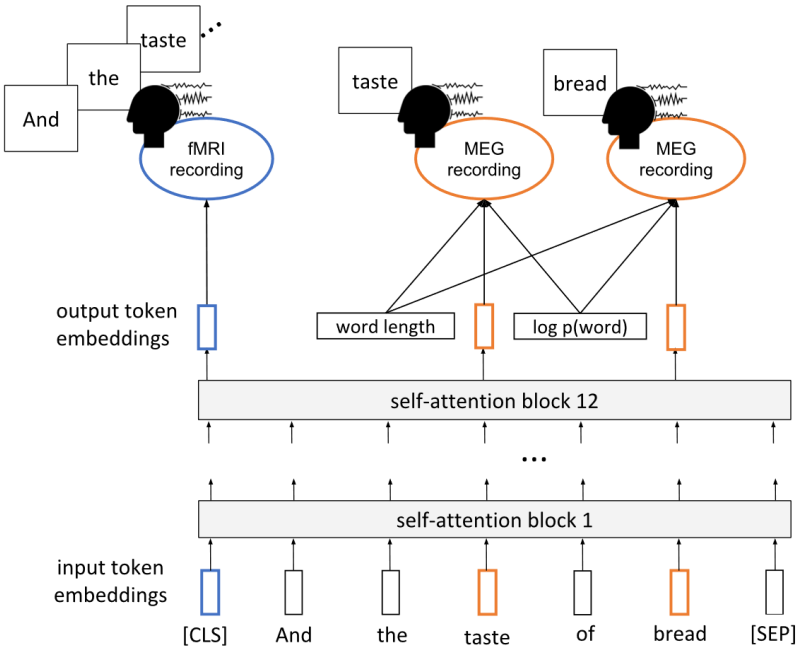


Figure 4.1: General approach for fine-tuning BERT using fMRI and/or MEG data. A linear layer maps the output token embeddings from the base architecture to brain activity recordings. Only MEG recordings that correspond to content words in the input sequence are considered. We include the word length and context-independent log-probability of each word when predicting MEG. fMRI data are predicted from the pooled embedding of the sequence, i.e. the [CLS] token embedding. For more details of the procedure, see section 4.3.2.

MEG was recorded from nine experiment participants using an Elekta Neuromag device (data for one participant had too many artifacts and was excluded, leaving 8 participants). This machine has 306 sensors distributed into 102 locations on the surface of the participant's head. The sampling frequency was 1kHz. The Signal Space Separation method (SSS) (Taulu, Kajola, and Simola, 2004) was used to reduce noise, and it was followed by its temporal extension (tSSS) (Taulu and Simola, 2006). The signal in every sensor was downsampled into 25ms non-overlapping time bins, meaning that each word in our data is associated with a 306 sensor $\times$ 20 time points image.

The fMRI data of nine experiment participants were composed of $3 \times 3 \times 3mm$ voxels. Data were slice-time and motion corrected using SPM8 (Kay et al., 2008). The data were then detrended in time and spatially smoothed with a $3mm$ full-width-half-max kernel. The brain surface of each subject was reconstructed using Freesurfer (Fischl, 2012), and a thick grey matter mask was obtained to select the voxels with neuronal tissue. For each subject, 50000-60000 voxels were kept after this masking. We use Pycortex (Gao et al., 2015) to handle and plot the fMRI data.

4.3.2 Model architecture

In our experiments, we build on the BERT architecture (Devlin et al., 2018), a specialization of a transformer network (Vaswani et al., 2017). Each block of layers in the network applies a transformation to its input embeddings by first applying self-attention (combining together the embeddings which are most similar to each other in several latent aspects). These combined embeddings are then further transformed to produce new features for the next block of layers. We use the PyTorch version of the BERT code provided by Hugging Face[2] with the pretrained weights provided by Devlin et al. (2018). This model includes 12 blocks of layers, and has been trained on the BooksCorpus (Zhu et al., 2015) as well as Wikipedia to predict masked words in text and to classify whether two sequences of words are consecutive in text or not. Two special tokens are attached to each input sequence in the BERT architecture. The [SEP] token is used

[2] https://github.com/huggingface/pytorch-pretrained-BERT/

to signal the end of a sequence, and the [CLS] token is trained to be a sequence-level representation of the input using the consecutive-sequence classification task. Fine-tuned versions of this pretrained BERT model have achieved state of the art performance in several downstream NLP tasks, including the GLUE benchmark tasks (Wang, Singh, et al., 2018). The recommended procedure for fine-tuning BERT is to add a simple linear layer that maps the output embeddings from the base architecture to a prediction task of interest. With this linear layer included, the model is fine-tuned end-to-end, i.e. all of the parameters of the model change during fine-tuning. For the most part, we follow this recommended procedure in our experiments. One slight modification we make is that in addition to using the output layer of the base model, we also concatenate to this output layer the word length and context-independent log-probability of each word (see Figure 4.1). Both of these word properties are known to modulate behavioral data and brain activity (Rayner, 1998; Van Petten and Kutas, 1990). When a single word is broken into multiple word-pieces by the BERT tokenizer, we attach this information to the first token and use dummy values (0 for word length and -20 for the log probability) for the other tokens. We use these same dummy values for the special [CLS] and [SEP] tokens. Because the time-resoluton of fMRI images is too low to resolve single words, we use the pooled output of BERT to predict fMRI data. In the pretrained model, the pooled representation of a sequence is a transformed version of the embedding of the [CLS] token, which is passed through a hidden layer and then a tanh function. We find empirically that using the [CLS] output embedding directly worked better than using this transformation, so we use the [CLS] output embedding as our pooled embedding.

4.3.3 Procedure

Input to the model. We are interested in modifying the pretrained BERT model to better capture brain-relevant language information. We approach this by training the model to predict both fMRI data and MEG data, each recorded (at different times from different participants) while experiment participants read a chapter of the same novel. fMRI records the blood-oxygenation-level dependent (BOLD) response, i.e. the relative amount of oxygenated blood in a given area of the brain, which is a function of how active the neurons are in that area of the brain. However, the BOLD response peaks 5 to 8 seconds after the activation of neurons in a region (Nishimoto et al., 2011; Wehbe, Murphy, et al., 2014; Huth et al., 2016). Because of this delay, we want a model which predicts brain activity to have access to the words that precede the timepoint at which the fMRI image is captured. Therefore, we use the 20 words (which cover the 10 seconds of time) leading up to each fMRI image as input to our model, irrespective of sentence boundaries. In contrast to the fMRI recordings, MEG recordings have much higher time resolution. For each word, we have 20 timepoints from 306 sensors. In our experiments where MEG data are used, the model makes a prediction for all of these $6120 = 306 \times 20$ values for each word. However, we only train and evaluate the model on content words. We define a content word as any word which is an adjective, adverb, auxiliary verb, noun, pronoun, proper noun, or verb (including to-be verbs). If the BERT tokenizer breaks a word into multiple tokens, we attach the MEG data to the first token for that word. We align the MEG data with all content words in the fMRI examples (i.e. the content words of the 20 words which precede each fMRI image).

Cross-validation. The fMRI data were recorded in four separate runs in the scanner for each participant. The MEG data were also recorded in four separate runs using the same division of the chapter as fMRI. We cross-validate over the fMRI runs. For each fMRI run, we train the model using the examples from the other three runs and use the fourth run to evaluate the model.

Preprocessing. To preprocess the fMRI data, we exclude the first 20 and final 15 fMRI images from each run to avoid warm-up and boundary effects. Words associated with these excluded images are also not used for MEG predictions. We linearly detrend the fMRI data within run, and standardize the data within run such that the variance of each voxel is 1 and the mean value of each voxel is 0 over the examples in the run. The MEG data is also detrended and standardized within fMRI run (i.e. within cross-validation fold) such that each time-sensor component has mean 0 and variance 1 over all of the content words in the run.

4.3.4 Models and experiments

In this study, we are interested in demonstrating that by fine-tuning a language model to predict brain activity, we can bias the model to encode brain-relevant language information. We also wish to show that the information the model encodes generalizes across multiple experiment participants, and multiple modalities of brain activity recording. For the current work, we compare the models we train to each other only in terms of how well they predict the fMRI data of the nine fMRI experiment participants, but in some cases we use MEG data to bias the model in our experiments. In all of our models, we use a base learning rate of 5×10^{-5}. The learning rate increases linearly from 0 to 5×10^{-5} during the first 10% of the training epochs and then decreases linearly back to 0 during the remaining epochs. We use mean squared error as our loss function in all models. We vary the number of epochs we use for training our models, based primarily on observations of when the models seem to begin to converge or overfit, but we match all of the hyperparameters between two models we are comparing. We also seed random initializations and allocate the same model parameters across our variations so that the initializations are consistent between each pair of models we compare.

Vanilla model. As a baseline, for each experiment participant, we add a linear layer to the pretrained BERT model and train this linear layer to map from the [CLS] token embedding to the fMRI data of that participant. The pretrained model parameters are frozen during this training, so the embeddings do not change. We refer to this model as the vanilla model. This model is trained for either 10, 20, or 30 epochs depending on which model we are comparing this to.

Participant-transfer model. To investigate whether the relationship between text and brain activity learned by a fine-tuned model transfers across experiment participants, we first fine-tune the model on the participant who had the most predictable brain activity. During this fine-tuning, we train only the linear layer for 2 epochs, followed by 18 epochs of training the entire model. Then, for each other experiment participant, we fix the model parameters, and train a linear layer on top of the model tuned towards the first participant. These linear-only models are trained for 10 epochs, and compared to the vanilla 10 epoch model.

Fine-tuned model. To investigate whether a model fine-tuned to predict each participant's data learns something beyond the linear mapping in the vanilla model, we fine-tune a model for each participant. We train only the linear layer of these models for 10 epochs, followed by 20 epochs of training the entire model.

MEG-transfer model. We use this model to investigate whether the relationship between text and brain activity learned by a model fine-tuned on MEG data transfers to fMRI data. We first fine-tune this model by training it to predict all eight MEG experiment participants' data (jointly). The MEG training is done by training only the linear output layer for 10 epochs, followed by 20 epochs of training the full model.

We then take the MEG fine-tuned model and train it to predict each fMRI experiment participant's data. This training also uses 10 epochs of only training the linear output layer followed by 20 epochs of full fine-tuning.

Fully joint model. Finally, we train a model to simultaneously predict all of the MEG experiment participants' data and the fMRI experiment participants' data. We train only the linear output layer of this model for 10 epochs, followed by 50 epochs of training the full model.

Evaluating model performance for brain prediction using the 20 vs. 20 test. We evaluate the quality of brain predictions made by a particular model by using the brain prediction in a classification task on held-out data, in a four-fold cross-validation setting. The classification task is to predict which of two sets of words was being read by the participant (Mitchell et al., 2008; Wehbe, Vaswani, et al., 2014; Wehbe, Murphy, et al., 2014). We begin by randomly sampling 20 examples from one of the fMRI runs. For each voxel, we take the true voxel values for these 20 examples and concatenate them together – this will be the target for that voxel. Next, we randomly sample a different set of 20 examples from the same fMRI run. We take the true voxel values for these 20 examples and concatenate them together – this will be our distractor. Next we compute the Euclidean distance between the voxel values predicted by a model on the target examples and the true voxel values on the target, and we compute the Euclidean distance between these same predicted voxel values and the true voxel values on the distractor examples. If the distance from the prediction to the target is less than the distance from the prediction to the distractor, then the sample has been accurately classified. We repeat this sampling procedure 1000 times to get an accuracy value for each voxel in the data. We observe that evaluating model performance using proportion of variance explained leads to qualitatively similar results (see Figure B.4), but we find the classification metric more intuitive and use it throughout the remainder of the chapter.

4.4 Results

Fine-tuned models predict fMRI data better than vanilla BERT. The first issue we were interested in resolving is whether fine-tuning a language model is any better for predicting brain activity than using regression from the pretrained BERT model. To show that it is, we train the fine-tuned model and compare it to the vanilla model by computing the accuracies of each model on the 20 vs. 20 classification task described in section 4.3.4. Figure 4.2 shows the difference in accuracy between the two models, with the difference computed at a varying number of voxels, starting with those that are predicted well by one of the two models and adding in voxels that are less and less well predicted by either. Figure 4.3 shows where on the brain the predictions differ between the two models, giving strong evidence that areas in the brain associated with language processing are predicted better by the fine-tuned models (Fedorenko and Thompson-Schill, 2014).

Relationships between text and brain activity generalize across experiment participants. The next issue we are interested in understanding is whether a model that is fine-tuned on one participant can fit a second participant's brain activity if the model parameters are frozen (so we only do a linear regression from the output embeddings of the fine-tuned model to the brain activity of the second participant). We call this the participant-transfer model. We fine-tune BERT on the experiment participant with the most predictable brain activity, and then compare that model to vanilla BERT. Voxels are predicted more accurately by the participant-transfer model than by the vanilla model (see Figure 4.2, lower left), indicating that we do get a transfer learning benefit.

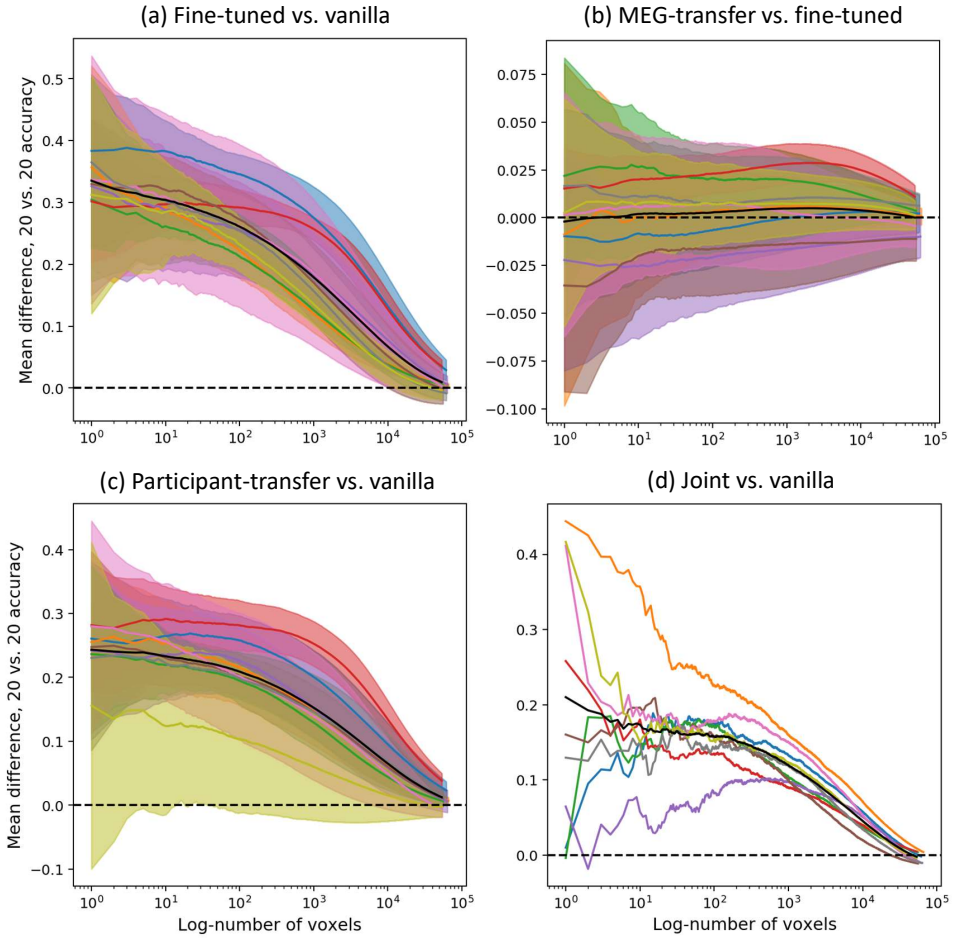


Figure 4.2: Comparison of accuracies of various models. In each quadrant of the figure above, we compare two models. Voxels are sorted on the x-axis in descending order of the maximum of the two models' accuracies in the 20 vs 20 test (described in section 4.3.4). The colored lines (one per participant) show differences between the two models' mean accuracies, where the mean is taken over all voxels to the left of each x-coordinate. In (a)-(c) Shaded regions show the standard deviation over 100 model initializations – that computation was not tractable in our framework for (d). The black line is the mean over all participants. In (a), (c), and (d), it is clear that the fine-tuned models are more accurate in predicting voxel activity than the vanilla model for a large number of voxels. In (b), the MEG-transfer model seems to have roughly the same accuracy as a model fine-tuned only on fMRI data, but in figure 4.3 we see that in language regions the MEG-transfer model appears to be more accurate.

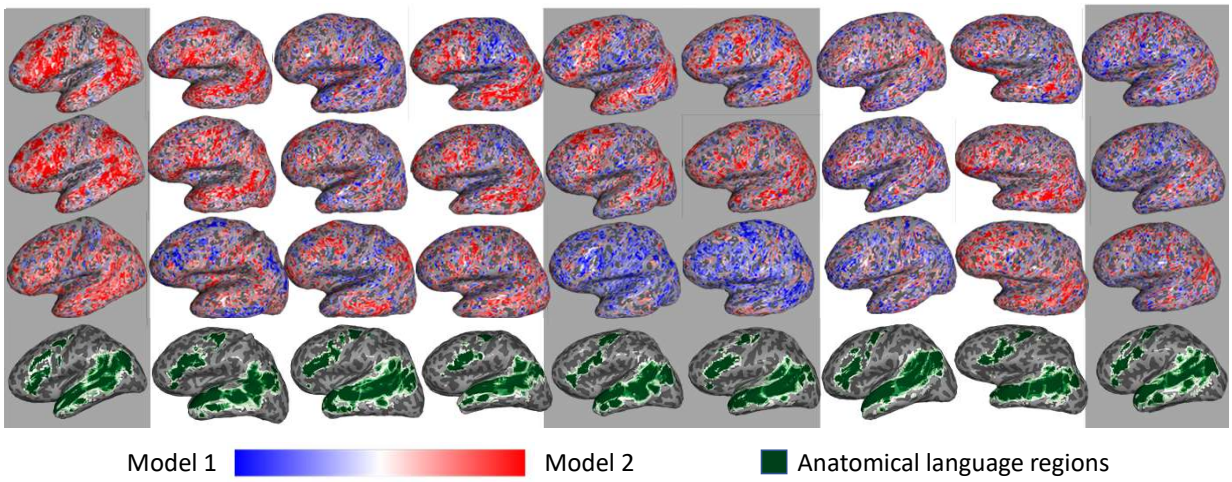


Figure 4.3: Comparison of accuracies on the 20 vs. 20 classification task (described in section 4.3.4) at a voxel level for all 9 participants we analyzed. Each column shows the inflated lateral view of the left hemisphere for one experiment participant. Moving from the top to third row, models 1 and 2 are respectively, the vanilla model and the fine-tuned model, the vanilla model and the participant-transfer model, and the fine-tuned model and MEG-transfer model. The leftmost column is the participant on whom the participant-transfer model is trained. Columns with a grey background indicate participants who are common between the fMRI and MEG experiments. Only voxels which were significantly different between the two models according to a related-sample t-test and corrected for false discovery rate at a .01 level using the Benjamini–Hochberg procedure (Benjamini and Hochberg, 1995) are shown. The color-map is set independently for each of the participants and comparisons shown such that the reddest value is at the $95th$ percentile of the absolute value of the significant differences and the bluest value is at the negative of this reddest value. We observe that both the fine-tuned and participant-transfer models outperform the vanilla model, especially in most regions that are considered to be part of the language network. As a reference, we show an approximation of the language network for each participant in the fourth row. These were approximated using an updated version of the Fedorenko, Hsieh, et al., 2010 language functional parcels[3], corresponding to areas of high overlap of the brain activations of 220 subjects for a "sentences>non-word" contrast. The parcels were transformed using Pycortex (Gao et al., 2015) to each participant's native space. The set of language parcels therefore serve as a strong prior for the location of the language system in each participant. Though the differences are much smaller in the third row than in the first two, we also see better performance in language regions when MEG data is included in the training procedure. Even in participants where performance is worse overall (e.g. the fifth and sixth columns of the third row), voxels where performance improves appear to be systematically distributed according to language processing function. Right hemisphere and medial views are available in appendix B.

Metric	Vanilla	MEG	Joint
CoLA	57.29	57.63	**57.97**
SST-2	93.00	**93.23**	91.62
MRPC (Acc.)	83.82	83.97	**84.04**
MRPC (F1)	88.85	**88.93**	88.91
STS-B (Pears.)	**89.70**	89.32	88.60
STS-B (Spear.)	**89.37**	88.87	88.23
QQP (Acc.)	90.72	**91.06**	90.87
QQP (F1)	87.41	**87.91**	87.69
MNLI-m	83.95	**84.26**	84.08
MNLI-mm	84.39	84.65	**85.15**
QNLI	89.04	**91.73**	91.49
RTE	61.01	**65.42**	62.02
WNLI	53.52	**53.80**	51.97

Table 4.1: GLUE benchmark results for the GLUE development sets. We compare the results of two of our models to the results published by https://github.com/huggingface/pytorch-pretrained-BERT/ for the pre-trained BERT model. The model labeled 'MEG' is the MEG transfer model described in section 4.3.4. The model labeled 'Joint' is the fully joint model also described in section 4.3.4. For all but one task, at least one of our two models is marginally better than the pretrained model. These results suggest that fine-tuning does not diminish – and possibly even enhances – the model's ability to perform NLP tasks.

Using MEG data can improve fMRI predictions. In a third comparison, we investigate whether a model can benefit from both MEG and fMRI data. We begin with the vanilla BERT model, fine-tune it to predict MEG data (we jointly train on eight MEG experiment participants), and then fine-tune the resulting model on fMRI data (separate models for each fMRI experiment participant). We see mixed results from this experiment. For some participants, there is a marginal improvement in prediction accuracy when MEG data is included compared to when it is not, while for others training first on MEG data is worse or makes no difference (see Figure 4.2, upper right). Figure 4.3 shows however, that for many of the participants, we see improvements in language areas despite the mean difference in accuracy being small.

A single model can be used to predict fMRI activity across multiple experiment participants. We compare the performance of a model trained jointly on all fMRI experiment participants and all MEG experiment participants to vanilla BERT (see Figure 4.2, lower right). We don't find that this model yet outperforms models trained individually for each participant, but it nonetheless outperforms vanilla BERT. This demonstrates the feasibility of fully joint training and we think that with the right hyperparameters, this model can perform as well as or better than individually trained models.

NLP tasks are not harmed by fine-tuning. We run two of our models (the MEG transfer model, and the fully joint model) on the GLUE benchmark (Wang, Singh, et al., 2018), and compare the results to standard BERT (Devlin et al., 2018) (see Table 4.1). These models were chosen because we thought they had the best chance of giving us interesting GLUE results, and they were the only two models we ran GLUE on. Apart from the semantic textual similarity (STS-B) task, all of the other tasks are very slightly improved on the development sets after the model has been fine-tuned on brain activity data. The STS-B task results are very slightly worse than the results for standard BERT. The fine-tuning may or may not be helping the model to perform these NLP tasks, but it clearly does not harm performance in these tasks.

Fine-tuning reduces [CLS] token attention to [SEP] token We evaluate how the attention in the model changes after fine-tuning on the brain recordings by contrasting the model attention in the fine-tuned and vanilla models described in section 4.3.4. We focus on the attention from the [CLS] token to other tokens

[3] https://evlab.mit.edu/funcloc/download-parcels

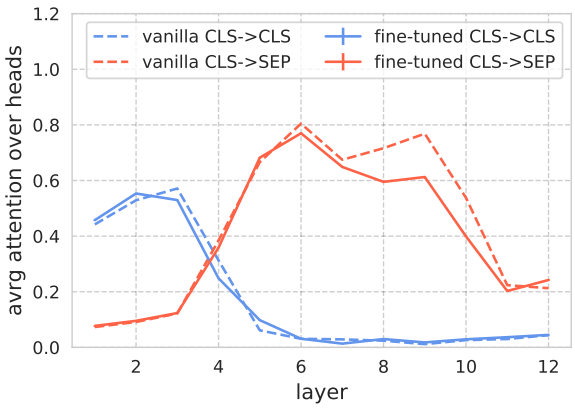


Figure 4.4: Comparison of attention from the [CLS] token to the [CLS] and [SEP] tokens between vanilla BERT and the fine-tuned BERT (mean and standard error over example presentations, attention heads, and different initialization runs). The attention from the [CLS] token noticeably shifts away from the [SEP] token in layers 8 and 9.

in the sequence because we use the [CLS] token as the pooled output representation of the input sequence. We observe that the [CLS] token from the fine-tuned model puts less attention on the [SEP] token in layers 8 and 9, when compared to the [CLS] token from the vanilla model (see Figure 4.4). Clark et al. (2019) suggest that attention to the [SEP] token in BERT is used as a no-op, when the function of the head is not currently applicable. Our observations that the fine-tuning reduces [CLS] attention to the [SEP] token can be interpreted in these terms. However, further analysis is needed to understand whether this reduction in attention is specifically due to the task of predicting fMRI recordings or generally arises during fine-tuning on any task.

Fine-tuning may change motion-related representations. In an effort to understand how the representations in BERT change when it is fine-tuned to predict brain activity, we examine the prevalence of various features in the examples where prediction accuracy changes the most after fine-tuning compared to the prevalence of those features in other examples. We score how much the prediction accuracy of each example changes after fine-tuning by looking at the percent change in Euclidean distance between the prediction and the target for our best participant on a set of voxels that we manually select which are likely to be language-related based on spatial location. We average these percent changes over all runs of the model, which gives us 25 samples per example. We take all examples where the absolute value of this average percent change is at least 10% as our set of changed examples, giving us 146 changed examples and leaving 1022 unchanged examples. We then compute the probability that each feature of interest appears on a word in a changed example and compare this to the probability that the feature appears on a word in an unchanged example, using bootstrap resampling on the examples with 100 bootstrap-samples to estimate a standard error on these probabilities. The features we evaluate come from judgments done by Wehbe, Murphy, et al., 2014 and are available online[4]. We examine all available features, but here we present only motion labels (Figure 4.6), emotion labels (Figure 4.7), and part-of-speech labels (Figure 4.8), as other features were either too sparse to evaluate or did not show any change in distribution. The sample sizes are relatively small in this analysis and should be viewed as preliminary, however, we see that examples containing verbs describing movement and imperative language are more prevalent in examples where accuracies change during fine-tuning. We believe the method of fine-tuning a model and evaluating feature distributions among the most changed examples is an exciting direction for future work.

[4] http://www.cs.cmu.edu/~fmri/plosone/

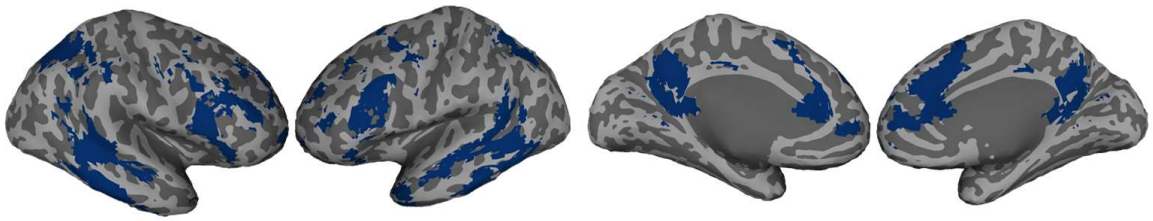


Figure 4.5: Voxels used to compute changes in accuracy between the fine-tuned and vanilla models for the feature distribution analysis described in Section 4.4. From left to right in the figure, are inflated lateral views of the right and left hemispheres followed by inflated medial views of the left and right hemispheres. Voxel selections are done manually based on location in the brain with the goal of restricting the accuracy computation to areas that are more likely to be involved in language processing.

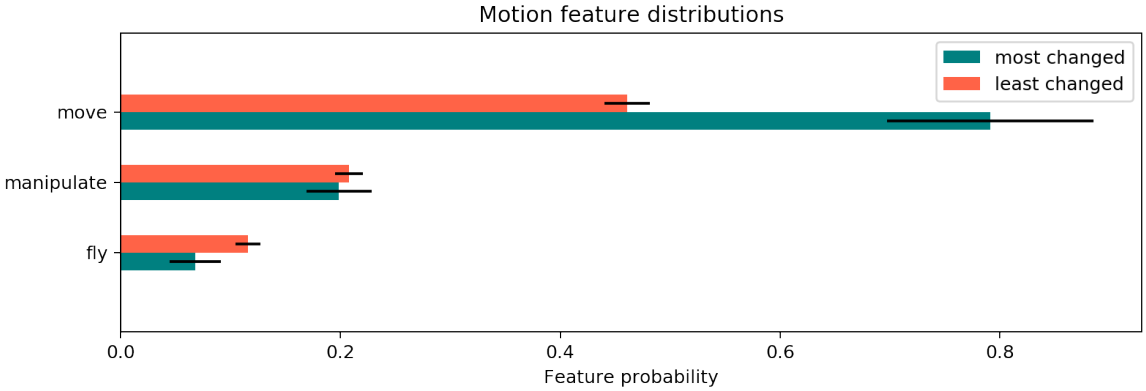


Figure 4.6: Prevalence of the motion-related labels on words in examples that are most and least changed in terms of prediction accuracy in language areas. The set of examples that is most and least changed is computed as described in Section 4.4. The most dramatic change in feature distribution among all features we examine, motion or otherwise, is the 'move' label.

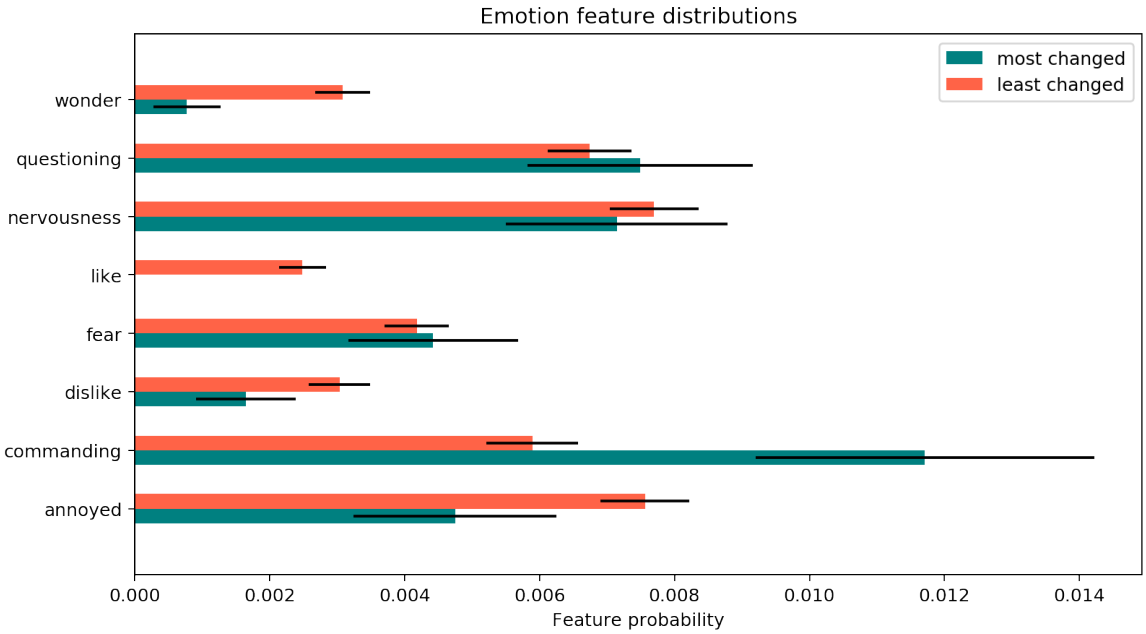


Figure 4.7: Prevalence of the emotion-related labels on words in examples that are most and least changed in terms of prediction accuracy in language areas. The set of examples that is most and least changed is computed as described in Section 4.4. There is some indication that representations for imperative language is changed during fine-tuning, as indicated by the change in prevalence of the 'commanding' label, but note that the prevalence is low for both the most changed and least changed examples, so this could easily be a sampling effect.

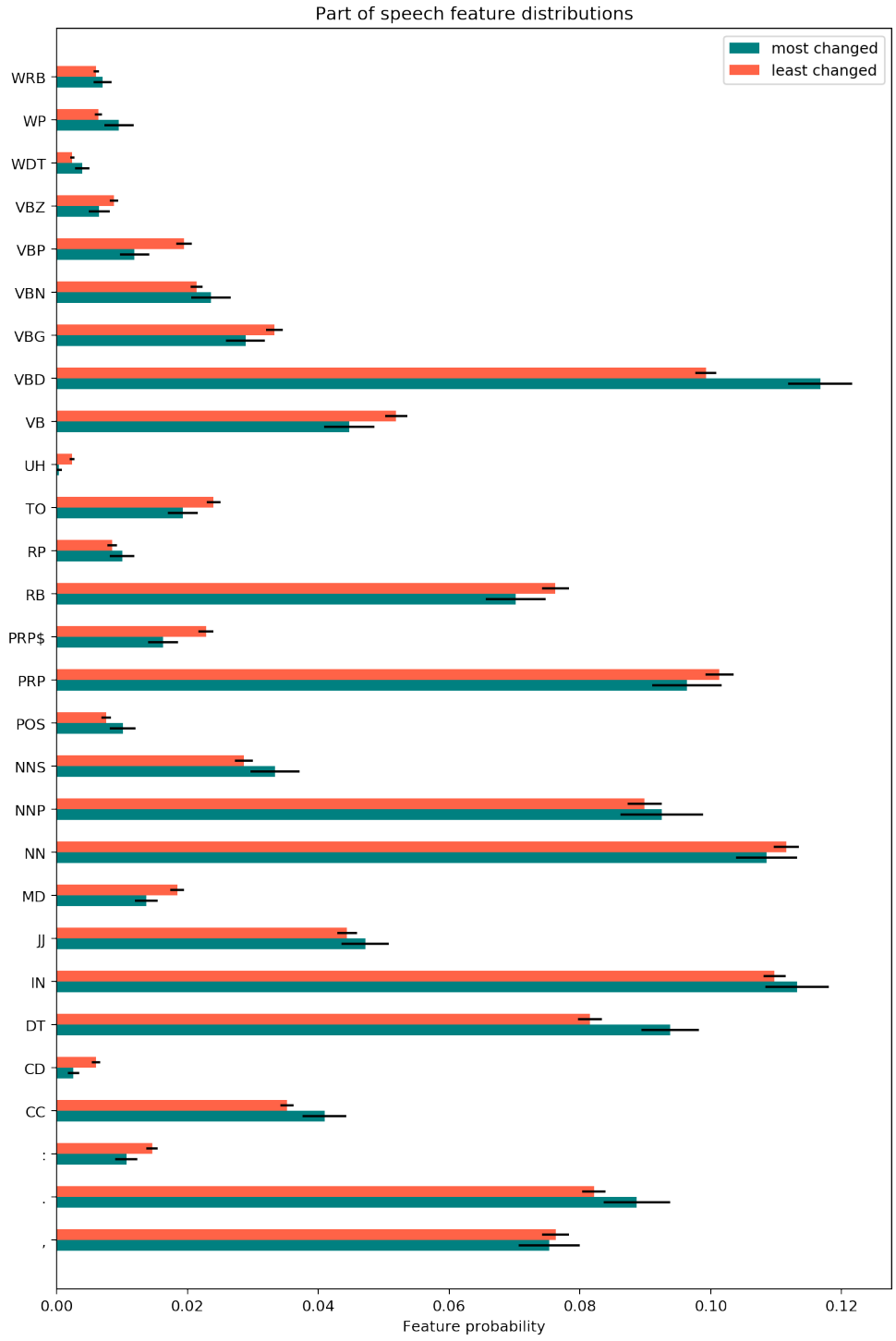


Figure 4.8: Prevalence of the part-of-speech labels on words in examples that are most and least changed in terms of prediction accuracy in language areas. The set of examples that is most and least changed is computed as described in Section 4.4. Verbs and determiners seem to be candidates for further study.

4.5 Discussion

This study aimed to show that it is possible to learn generalizable relationships between text and brain activity by fine-tuning a language model to predict brain activity. We believe that our results provide several lines of evidence that this hypothesis holds.

First, because a model which is fine-tuned to predict brain activity tends to have higher accuracy than a model which just computes a regression between standard contextualized-word embeddings and brain activity, the fine-tuning must be changing something about how the model encodes language to improve this prediction accuracy.

Second, because the embeddings produced by a model fine-tuned on one experiment participant better fit a second participant's brain activity than the embeddings from the vanilla model (as evidenced by our participant-transfer experiment), the changes the model makes to how it encodes language during fine-tuning at least partially generalize to new participants.

Third, for some participants, when a model is fine-tuned on MEG data, the resulting changes to the language-encoding that the model uses benefit subsequent training on fMRI data compared to starting with a vanilla language model. This suggests that the changes to the language representations induced by the MEG data are not entirely imaging modality-specific, and that indeed the model is learning the relationship between language and brain activity as opposed to the relationship between language and a brain activity *recording modality*.

Models which have been fine-tuned to predict brain activity are no worse at NLP tasks than the vanilla BERT model, which suggests that the changes made to how language is represented improve a model's ability to predict brain activity without doing harm to how well the representations work for language processing itself. We suggest that this is evidence that the model is learning to encode brain-activity-relevant language information, i.e. that this biases the model to learn representations which are better correlated to the representations used by people. It is non-trivial to understand exactly how the representations the model uses are modified, but we investigate this by examining how the model's attention mechanism changes, and by looking at which language features are more likely to appear on examples that are better predicted after fine-tuning. We believe that a more thorough investigation into how model representations change when biased by brain activity is a very exciting direction for future work.

Finally, we show that a model which is jointly trained to predict MEG data from multiple experiment participants and fMRI data from multiple experiment participants can more accurately predict fMRI data for those participants than a linear regression from a vanilla language model. This demonstrates that a single model can make predictions for all experiment participants – further evidence that the changes to the language representations learned by the fine-tuned model are generalizable. There are optimization issues that remain unsolved in jointly training a model, but we believe that ultimately it will be a better model for predicting brain activity than models trained on a single experiment participant or trained in sequence on multiple participants.

4.6 Conclusion

Fine-tuning language models to predict brain activity is a new paradigm in learning about human language processing. The technique is very adaptable. Because it relies on encoding information from targets of a prediction task into the model parameters, the same model can be applied to prediction tasks with different sizes and with varying temporal and spatial resolution. Additionally it provides an elegant way to leverage massive data sets in the study of human language processing. To be sure, more research needs to be done on how best to optimize these models to take advantage of multiple sources of information about language

processing in the brain and on improving training methods for the low signal-to-noise-ratio setting of brain activity recordings. Nonetheless, this study demonstrates the feasibility of biasing language models to learn relationships between text and brain activity. We believe that this presents an exciting opportunity for researchers who are interested in understanding more about human language processing, and that the methodology opens new and interesting avenues of exploration.

Chapter 5

Relating Cognitive Processes in the Brain to NLP Tasks Using Model Parameter Similarity

In this chapter, we use a method, different from the method in chapter 3, for analyzing task similarity in a multitask setting. Here, we use a bottleneck layer to constrain a deep language model to learn a small number of latent functions of an input sequence of text which can support multiple task predictions. We suggest that by characterizing the degree to which the model relies on the same latent functions when performing one task prediction as when performing others, we can gain insight both into how the model is making its predictions and into the similarity of the tasks in terms of those mechanisms. Whereas the method in chapter 3 produces binary task similarities indicating only whether tasks are related, the task similarities we produce in this chapter characterize degree of similarity and thus produce more interpretable results. Additionally, the method in chapter 3 requires retraining the model for each pair of tasks, but in this chapter, we train the model once with all tasks (but in both chapters variance in the results is assessed by multiple model initializations and trainings). In the current chapter, we apply our similarity method on two sets of tasks. The first set of tasks are standard natural language processing (NLP) tasks, such as part-of-speech prediction and semantic role labeling. We characterize how similar these tasks are to each other using the method we develop, and discuss how these task similarities give us insight into the mechanisms that the model uses to make its predictions. In addition to the NLP tasks, we apply our method to a second set of tasks that we refer to as cognition-relevant. These cognition-relevant tasks are prediction of brain activity recordings (fMRI and EEG-derived event related potential (ERP) data) as well as prediction of behavioral data (eye-tracking and self-paced reading times) all recorded while participants read text. By using task similarity on these tasks we can gain insight into how the model is making its predictions of cognition-relevant task outputs. We suggest that understanding of how the model makes predictions of cognition-relevant tasks can ultimately lead to insights about the cognitive processes driving the measurements the model predicts. We find that ERP data prediction is similar to the prediction of patient-like semantic roles, while eye-tracking and self-paced reading data prediction is similar to the prediction of modifiers/modified items, punctuation, and multiword expressions which might induce eye-movements and/or integration operations. Lastly we find that fMRI prediction is somewhat similar to patient-like and agent-like decompositional semantics along with sentence length, but that it has low overlap with most of the tasks in our model. Our findings are evidence for semantics being the main driver of cognition-relevant task data, but more importantly we believe our method can be applied to continually refine hypotheses about cognition in the brain.

5.1 Introduction

In this chapter, we use multitask learning to train a single model which takes as input a sequence of text, and produces as output predictions of natural language processing (NLP) task labels, such as part-of-speech, along with predictions of brain activity (fMRI images, EEG data), and behavioral data (eye-tracking data and self-paced reading times). We refer collectively to the brain activity recordings and behavioral data as cognition-relevant tasks. These data are all collected from participants reading either sentences or narratives. We constrain the model with a bottleneck layer, forcing it to learn a small number of latent functions that produce a small, shared latent representation from which all of the output tasks are predicted. We use the similarity in the mapping (weights) from this shared latent representation to individual task outputs as a measure of task similarity. Under this definition of task similarity, we are comparing which latent functions support each task, rather than directly comparing representations of the input text across tasks. Because the method does not directly compare representations, we do not need to record brain activity on the same corpora which have NLP labels in order to compare brain activity prediction to the prediction of those labels. This is a major advantage over simple correlation or representational similarity analysis (RSA). In the case of brain activity and behavioral data, we suggest that the functions that the model learns between the input text and the output predictions capture information that is relevant to the cognitive processes involved in language processing. This assumption is supported by the work in chapter 4, which shows that when a model is fine-tuned to predict the fMRI recordings of one person, that model generalizes to predict the fMRI recordings of others better than a baseline model, and that to some extent transfer is also possible from MEG to fMRI. Those findings suggest that a fine-tuned model learns to capture information about human language processing itself — not just information which is specific to a particular person's brain activity or to a particular recording modality — in its parameters. The method we describe in this chapter enables an exploration of the mechanisms a model uses to make its predictions. In and of itself, that is useful for better understanding black box deep language models and what they do. Additionally, the comparison of cognition-relevant, but inscrutable tasks such as prediction of the LAN event-related potential, to more interpretable NLP tasks in terms of how the model is making its predictions also enables us to quantify the extent to which the cognition-relevant prediction relies on features which correlate with, for example, specific syntactic constructions or semantic information. Placing the cognition-relevant task predictions in this context gives us insight into the cognitive processes underlying observations such as the LAN. For example, we find that all event-related potentials are related to patient-like semantics more than to syntax. Here we apply our method with a set of roughly eighty tasks and show that it can be used to make inferences about the cognitive processes underlying behavior and brain activity.

5.2 Related Work

Task similarity. Multitask learning has often been used in the context of exploiting shared regularities across tasks to improve the generalization error of a trained model and reduce the amount of labeled data required to solve the individual tasks. Early work on these techniques argued that in the hypothesis space given to a learning method, where the learning method's goal is to choose the hypothesis that best fits the data distribution, some hypotheses will work for all tasks while others will work for only some tasks, and the hypotheses that work for all tasks are more likely to fit the true data distribution, i.e. have lower generalization error (Caruana, 1997; Baxter, 2000). This idea has been theoretically studied, especially in the related field of domain adaptation, where theoretical bounds on generalization error have been developed based on the intuition that for transfer learning to be successful, the source and target domain need to be similar. Task-relatedness has therefore been characterized using distances between

distributions, where the sample points for a task t, $\mathcal{S}_t = \{(x_1, y_1), ..., (x_n, y_n)\} \in (X \times Y)^n$ are drawn from some underlying distribution Q_t, and tasks are compared using a distance measure $d(Q_i, Q_j)$ on these distributions (Ben-David and Schuller, 2003; Ben-David, Blitzer, Crammer, and Pereira, 2007; Ben-David, Blitzer, Crammer, Kulesza, et al., 2010; Mansour, Mohri, and Rostamizadeh, 2009; Germain et al., 2013). The distance functions used in the development of the theory are typically not computable in practice, but can be used to derive bounds that can be optimized, e.g. (Germain et al., 2013), or to justify algorithms. One such algorithm that has been recently applied in deep networks is domain adversarial networks (Ganin et al., 2016), which adds a label to the output indicating which task the network is performing, and reverses the gradient on that label. This encourages the network to learn features which cannot discriminate which task is being performed (i.e. the distribution of the inputs to the decoder part of the network should be the same for all tasks). The task sampling method we use (see section 5.3.4) can be seen as a relaxed version of this adversarial algorithm, since it downweights a task if the gradient on that task's loss with respect to the common parameters of the model is different from the average gradient with respect to the common parameters over all tasks. A related idea, going back 25 years, is to learn a distance metric on binary classification tasks which uses Euclidean distance on the input features, but reweights those features such that the distance between tasks is minimized on examples where the labels are the same and maximized on samples where the labels are different (Thrun and O'Sullivan, 1996). Task-relatedness has also been implicitly characterized in terms of parameter similarity. Many regularization strategies have been developed for multitask learning, mostly in the linear model setting, which implicitly or explicitly impose a prior on parameterized functions. These regularization strategies bias the learning to prefer that similar tasks have similar parameters. Some of these use group sparsity or similar structure on the input features to force similar tasks to use similar support (Kim and Xing, 2010; Chen et al., 2010), some cluster tasks together (Jacob, Vert, and Bach, 2009), and some draw the parameters from a distribution with various characteristics (Evgeniou, Micchelli, and Pontil, 2005; Zhang and Yeung, 2012). Additionally, the task-relationship matrix can be learned, and used to influence how parameters are updated, for example in (Saha et al., 2011). In some models, e.g. (Zhang and Yeung, 2012), the covariance of the parameters is controlled by a prior distribution which is itself parameterized by the task similarity matrix — in other words the covariance is explicitly related to task similarity. In this context, our use of (modified) cosine similarity between task weights can be seen as a natural measure of task similarity.

Deep learning, multitask learning, and representational similarity in computational cognitive neuroscience. Recent work in computational cognitive neuroscience has leveraged deep learning as part of the toolbox for analyzing cognitive processes. Existing work leverages deep models through correlation of the feature activations of off-the-shelf deep models to brain activity (Yamins et al., 2014; Schrimpf et al., 2018), the ability of an off-the-shelf model to predict brain activity and behavior (Wehbe, Vaswani, et al., 2014; Hollenstein et al., 2019), interpretable functions of the representations in off-the-shelf models (Hale et al., 2018), manipulation of inputs to the models to explore the effect on brain activity prediction (Jain and Huth, 2018; Toneva and Wehbe, 2019), and fine-tuning deep models to predict brain activity and behavioral measures (Schwartz, Toneva, and Wehbe, 2019; Schwartz and Mitchell, 2019). In an orthogonal line of work, representational similarity analysis (RSA) (Kriegeskorte, Mur, and Bandettini, 2008) has been a very popular tool for exploring how the brain represents stimuli. RSA utilizes (dis)similarity matrices in which each element gives a score for the (dis)similarity of the representation of a pair of stimuli under either a model or a recording of brain activity. These (dis)similarity matrices can then be compared across models (typically a model's representation is compared to the brain's representation) to score how similar the representation spaces are. However a major drawback of RSA is that it requires either brain activity recordings for the same set of stimuli that the model has been exposed to, or it requires running the model

on a set of stimuli that brain activity recordings already exist for, but that it is not necessarily tuned to represent well.

Here, we propose an alternative approach for leveraging deep learning models to learn about cognitive processes. Namely, we propose to use a multitask paradigm in which we jointly train a model on datasets that do not necessarily have overlapping examples, encouraging the model to learn representations of the input that enable it to perform all of the different tasks well. Then, we use the parameters of the model to measure the similarity across different tasks, bypassing the need to have common examples across datasets. Like RSA, this gives us a score of how similar the representations in cognition-relevant tasks (brain activity prediction and behavioral data prediction) are to the representations used in other tasks, but our method can leverage pre-existing labeled datasets without requiring new brain activity recordings, and we can be assured that the model has good representations for the stimuli in existing brain activity recordings. The model weights tell us which aspects of the shared latent space are most important to a particular task, and thus comparison of the model weights can be viewed as an indirect comparison of the representation spaces of different tasks, without requiring tasks to share stimuli (input examples).

Several authors have considered scoring the similarity in representations between brain activity and deep learning models. For example, Toneva and Wehbe (2019) and Schrimpf et al. (2018) emphasize evaluating deep learning models using brain activity by exploring how context modifies the representations in a model and how this relates to brain activity, or how the feature activations in a model compare to representations in the brain. Utilizing multiple tasks as a means of interpreting the cognitive processes underlying human vision and language understanding has also been explored. Wang, Tarr, and Wehbe (2019) utilizes a set of models trained for specific tasks as a basis for predicting brain activity and analyzes task similarity based on the cosine similarity of which voxels each task predicts well. The mapping of the tasks onto voxels in the brain also enables inference of how the cognitive processes in certain regions share information with a task. Çukur et al. (2013) examines how the weights mapping from an input to brain activity change as a function of attention, notably using correlation between weights as a similarity measure. Schwartz and Mitchell (2019) uses fine-tuning of a deep learning model with different combinations of event-related potential (ERP) component prediction tasks in an attempt to find which ERP components share information with each other. The current work fits into this context in that it uses high-level interpretable tasks (NLP tasks) as a way to infer some of the cognitive processing that drives brain activity and behavior (eye-tracking, self-paced reading) while a person reads text, but stands apart in its use of model weights to measure task similarity. This strategy allows us to take advantage of existing large labeled datasets to learn about the cognitive processes in the brain.

5.3 Methods

5.3.1 Tasks

The task outputs we train our model to predict consist of measurements of brain activity, behavioral measurements, and standard natural language processing (NLP) task outputs. The NLP tasks that we include consist of tasks which relate to the complexity of the input sequence of text, the syntax of the input sequence, and the semantics of the input sequence. We use the NLP tasks as a way to interpret what information subserves the cognitive processes that generate the brain activity and behavioral measurements we can observe. In this section we give a brief overview of the tasks. More details can be found in appendix C.

Cognition-relevant tasks. We include two types of brain activity tasks and two kinds of behavioral data tasks. The first kind of brain activity data we include is data recorded by functional magnetic resonance imaging (fMRI) (Wehbe, Murphy, et al., 2014)[1] while participants read a chapter of *Harry Potter and the Sorcerer's Stone* (Rowling, 1999). The chapter is shown to each participant word-by-word every 0.5 seconds. fMRI records the blood oxygenation level dependent (BOLD) response. The oxygenation level in the blood changes near active neurons in response to the demands of those neurons, but the change is slow and follows a stereotypical curve called the hemodynamic response function (HRF) that peaks around 8 seconds after the the neuronal activity (Kemmerer, 2014). Because of this and the slow timing of fMRI images (one image is produced every two seconds), we treat fMRI prediction as a sequence-level task in which we predict each fMRI image as a function of the sequence of the 20 words leading up to the time at which that image was recorded. The fMRI data are recorded from 9 participants, and we predict each participant's data as a separate task.

The second type of brain activity data we include is derived from electroencephalography (EEG) by taking the average of a predefined set of EEG electrodes during a predefined window of time to measure stereotyped responses to stimuli known as event-related potentials (ERPs). The data is recorded on 205 sentences sampled from 3 novels in British English (the University College London (UCL) corpus) (Frank, Otten, et al., 2015). We average together the data from the 24 participants, so that we have one scalar value per word per ERP component task (six tasks).

The first kind of behavioral data we include is self-paced reading time (the participant must press a button to advance to the next word and the timing of the button presses is recorded). Our self-paced reading time task also comes from the UCL corpus and uses 361 sentences (these include the same 205 sentences as the ERP data) (Frank, Monsalve, et al., 2013). We average together the self-paced reading times across participants to give us a single scalar value per word.

Finally, the other kind of behavioral data we include is eye-tracking data. We use three datasets for eye-tracking data. One is the UCL corpus, which uses the same 205 sentences as the ERP data (Frank, Monsalve, et al., 2013), one is the Ghent Eye Tracking Corpus (GECO) (Cop et al., 2017), which has data for an entire novel, and one is the Dundee corpus, which has data for excerpts from a newspaper (Kennedy, Hill, and Pynte, 2003). Additional information about preprocessing, train-test splits, the specific eye-tracking measures we include, and other details about all of our tasks are available in appendix C.

Natural Language Processing (NLP) tasks. The set of natural language processing tasks we include are drawn from three different sets of prior work: we include prediction on the Stanford Sentiment Treebank as a task (Socher et al., 2013), the suite of tasks from Conneau et al. (2018), and the suite of tasks from Tenney et al. (2019). The tasks can be broadly categorized as relating to the complexity of the input sequence of text (sentence length and parse tree depth), the syntax of the input (prediction of part-of-speech, constituent, dependency role, bshift, coordination-inversion, object number, subject number, tense, and top-constituents), and the semantics of the input (semantic role labeling, semantic proto-role labeling (36 tasks), named entity recognition, coreference prediction, SemEval 2010 Task 8 relations, definite-pronoun resolution, sentiment prediction, and semantic odd man out). For short descriptions of each of the tasks see table 5.2, and for detailed information about the tasks we refer the reader to the original papers. These datasets all have predefined train-test splits. We reserve 20% of the training set to adjust how frequently we sample tasks (the examples selected for this task-sample-rate tuning set are constant across model initializations), and use the remaining part of the training sets for primary training while evaluating on the predefined validation sets.

[1] available at http://www.cs.cmu.edu/~fmri/plosone/

Dataset	Description	Task	Examples
Harry Potter (Wehbe, Murphy, et al., 2014)	A chapter from *Harry Potter and the Sorcerer's Stone* (Rowling, 1999)	fMRI	1.2K
UCL (Frank, Monsalve, et al., 2013; Frank, Otten, et al., 2015)	Collection of sentences from 3 novels	ERP eye-tracking (UCL) self-paced reading time	2K 2K 5K
GECO (Cop et al., 2017)	*The Mysterious Affair at Styles* by Agatha Christie (Christie, 1920) (entire novel)	eye-tracking (GECO)	56K
Dundee (Kennedy, Hill, and Pynte, 2003)	Collection of text from *The Independent* (British newspaper)	eye-tracking (Dundee)	52K
Stanford Sentiment Treebank (Socher et al., 2013)	Collection of movie reviews from rottentomatoes.com	sentiment	11K
Toronto Books Corpus (Zhu et al., 2015)	Sentences from 11,038 books, sampled for each task (described in Conneau et al. (2018))	bshift coord-inv object number semantic odd man out subject number tense sentence length top constituents tree depth	120K 120K 120K 120K 120K 120K 120K 120K 120K
OntoNotes (Weischedel et al., 2011)	Sentences from newswire	part of speech constituent named entity coreference semantic role	2.5M 2M 161K 321K 738K
English Web Treebank (Silveira et al., 2014)	English web portion of Universal Dependencies 2.2 release (sentences from the internet)	dependency role proto-roles 2: (awareness, volition, ...)	254K 6K
SemEval (Hendrickx et al., 2010)	Collection of sentences from the internet used for SemEval 2010 Task 8	SemEval	10.7K
DPR (Rahman and Ng, 2012)	Hand crafted sentences created for ambiguous coreference	definite-pronoun resolution	3.1K
PropBank(Palmer, Gildea, and Kingsbury, 2005)	Penn Treebank sentences from *The Wall Street Journal*	proto-roles 1: (awareness, volition, ...)	9.8K

Table 5.1: The datasets used for our tasks. Examples refers to the total number of input-output pairs for a task. Note the heterogeneity in content (fiction, narrative, news, web), number of examples (1.2K-2.5M), and labeling (binary class, multiclass, real number).

Task	Suite	Description	Type	Classes
fMRI	Harry Potter	Predict the value for each fMRI voxel	sequence	n/a
ERP	UCL	Predict the value for each ERP component (ELAN, LAN, N400, P600, EPNP, PNP)	word	n/a
eye-tracking (UCL)	UCL	Predict each eye-tracking measure (first-fixation, first-pass, right-bounded, go-past)	word	n/a
eye-tracking (GECO)	GECO	Predict each eye-tracking measure (first-fixation, first-pass, fixation-count, go-past, total-reading-time)	word	n/a
eye-tracking (Dundee)	Dundee	Predict each eye-tracking measure (first-pass, go-past, right-bounded)	word	n/a
self-paced reading time	UCL	Predict the self-paced reading time measure	word	n/a
sentiment	SST	Does the movie review have positive sentiment?	sequence	2
bshift	cram	Does an inverted bigram exist?	sequence	2
coord-inv	cram	Have a coordinate-clause's subclauses been swapped?	sequence	2
object number	cram	Is the object in the sentence plural?	sequence	2
semantic odd man out	cram	Has a word been changed to a different word?	sequence	2
subject number	cram	Is the subject in the sentence plural?	sequence	2
tense	cram	Is the verb past tense?	sequence	2
sentence length	cram	Quantized sentence length	sequence	6
top constituents	cram	Predict which of the 20 most common tuple of constituents is just below the S node in this sentence	sequence	20
tree depth	cram	Quantized tree depth	sequence	7
part-of-speech	edge	Part of speech for a word	word	48
constituent	edge	Constituent for a phrase	1-span	30
named entity	edge	Named entity recognition	1-span	18
coreference	edge	Coreference prediction	2-span	2
semantic role	edge	Semantic role of an argument in a predicate	2-span	66
dependency role	edge	Dependency label of an edge in a dependency tree	2-span	49
SemEval	edge	Semantic relation between a pair of nouns	2-span	19
Definite-pronoun resolution	edge	Are the spans coreferent?	2-span	2
awareness	edge (pr1/pr2)	Arg. is aware of being involved in pred.	2-span	2
change of location	edge (pr1/pr2)	Arg. changed location during pred.	2-span	2
change of state	edge (pr1/pr2)	Arg. was altered by end of pred.	2-span	2
change of state continuous	edge (pr2)	Change in arg. happened throughout pred.	2-span	2
changes poss.	edge (pr1/pr2)	Arg. changed possession during pred.	2-span	2
existed after	edge (pr1/pr2)	Arg. existed after pred. stopped	2-span	2
existed before	edge (pr1/pr2)	Arg. existed before pred. began	2-span	2
existed during	edge (pr1/pr2)	Arg. existed during pred.	2-span	2
exists as physical	edge (pr1/pr2)	Arg. existed as a physical object	2-span	2
instigation	edge (pr1/pr2)	Arg. caused pred. to happen	2-span	2
location of event	edge (pr1/pr2)	Arg. describes the location of pred.	2-span	2
makes physical contact	edge (pr1/pr2)	Arg. made physical contact with someone or something else involved in pred.	2-span	2
partitive	edge (pr2)	Only a part of arg. was involved in pred.	2-span	2
manipulated by	edge (pr1/pr2)	Arg. is physically manipulated by someone or something else involved in pred.	2-span	2
predicate changes arg	edge (pr1/pr2)	predicate caused a change in arg.	2-span	2
sentient	edge (pr1/pr2)	Arg. is sentient	2-span	2
stationary	edge (pr1/pr2)	Arg was stationary during pred.	2-span	2
volition	edge (pr1/pr2)	Arg. chose to be involved in pred.	2-span	2
was for benefit	edge (pr2)	Pred. happened for the benefit of arg.	2-span	2
was used	edge (pr2)	Arg. was used in carrying out pred.	2-span	2

Table 5.2: Description of tasks included in our setup. The type column indicates what kind of prediction we make for a task. Sequence means there is a single prediction for a sequence of words, word means there is a prediction for each word, 1-span means for each example a single span of words is predefined in the data, and we make a prediction for this span, and 2-span means for each example two spans of words are predefined in the data and we make a single prediction for the two spans jointly. For example, 2-spans define the dependency role relation task, in which each relation we try to predict takes two arguments and the spans identify those arguments. When the classes column has an n/a entry, that task is a regression task.

5.3.2 Model architecture and task weight vectors

Our model architecture consists of a single shared encoder and one decoder for each task we predict. Architecturally, we have four different kinds of tasks (see figure 5.1) — tasks defined at the sequence level, tasks defined at the word level, tasks defined at the span level (i.e. a contiguous span of words which is shorter than the input sequence), and fMRI prediction (a special kind of sequence-level task which we treat separately in the architecture). For word-level and sequence-level tasks (excluding fMRI), each decoder is simply a linear layer which maps from the appropriate bottleneck embedding produced by the encoder to the output prediction for a task. For span-level and fMRI tasks, we create a simple intermediate embedding as discussed below, and map from that intermediate embedding to the output. Any of the word-level, sequence-level, and span-level tasks may also be a multiclass task. In the case of a multiclass task, our decoder has a weight vector mapping from the appropriate embedding (e.g. the bottleneck embedding associated with a word for a word-level task) to each of the individual candidate classes. The decoder then chooses among the candidate classes by applying a softmax over them. For the purpose of computing task similarity, we consider each of the weight vectors for each of the individual classes as a separate "sub-task". For example, we compare the prediction of the label *NNP* (proper-noun) in the part-of-speech task to every other task-associated weight vector in our model.

Shared encoder. We build our encoder around the `bert-base-uncased` variant of the BERT language model (Devlin et al., 2018) using the code provided by Hugging Face [2] and the pretrained weights provided by Devlin et al. (2018). For each token in a sequence, we concatenate the output of each of the twelve self-attention blocks in BERT for that token along with the input token embedding, the word length, and the log-probability of a word. Following Schwartz, Toneva, and Wehbe (2019), when a word is broken into multiple tokens, the first token gets the value for the word length and log-probability of the word, and the other tokens get dummy values for those components (0 and -20 respectively). We use the same dummy values for the special [CLS] and [SEP] tokens ([CLS] is a special token used in BERT for sequence-level representations, which is always prepended to the input, and [SEP] is a special token used in BERT to designate the end of a sequence). The concatenation of the BERT outputs and the word attributes gives us a vector of $768 \times 13 + 2 = 9986$ components per token, where 13 comes from the 12 self-attention blocks in the BERT base model plus the input embedding, and 768 is the number of components in each BERT self-attention block output representation. We pass this concatenated vector through a linear layer to produce a vector of 30 components, which is then modified by a Gaussian error linear unit (GELU) activation function (Hendrycks and Gimpel, 2016), followed by layer normalization (Ba, Kiros, and Hinton, 2016) to produce our bottleneck representation of size 30. This representation is shared across all tasks, and the design principle we follow is to try to do as little as possible in the model after this layer, so that we can think of all of the parts of the model between the input sequence of text and this bottleneck embedding (the encoder) as learning 30 latent functions over the input text which can support all of our different tasks, and we can compare our tasks in terms of which latent functions each task relies on.

Word-level and sequence-level tasks are predicted directly. For a word-level prediction task such as an ERP, eye-tracking, or self-paced reading task, we directly predict an output value from the bottleneck representation associated with the appropriate word. If a word breaks into multiple tokens, the prediction is made from the bottleneck embedding associated with the first token. For sequence-level predictions (excluding fMRI predictions) we predict the output value directly from the bottleneck representation associated with the [CLS] token (the bottleneck representation associated with the sequence).

[2] `https://huggingface.co/transformers/index.html`

Span-level tasks use max-pooling. For span-level tasks, the span of text that the model is meant to label is given to the model in the input. We form a single representation for the span by taking the element-wise maximum over the bottleneck embeddings associated with each token in the span, then we predict the output task from that span embedding. The weight vectors we associate with span-level tasks map from this span embedding to the task outputs. Some span-level tasks actually identify two spans in the input. For example, for the semantic role labeling task, we have a span which identifies the argument the model needs to label, and a span which identifies the verb context with respect to which the model is labeling the argument. We make a prediction for these two-span tasks by concatenating the individual span embeddings together and making a prediction from the concatenated embedding. This is equivalent to having a weight vector associated with each of the individual spans and then adding together the outputs of the inner products of those individual weight vectors and spans, so in our similarity computation we consider the weight vectors for each of the spans separately. Since semantic role labeling is a multiclass task, we thus have a weight vector associated with each individual class for each of the two spans. So, we can take one of the classes of this task, e.g. ARG0, and we can look at how the weight vector associated with the verb context span for the ARG0 class prediction compares to every other weight vector in our model. Similarly, we can look at how the weight vector associated with the argument span for the ARG0 class prediction compares to every other weight vector in our model.

fMRI tasks use a special HRF embedding. For fMRI tasks, we have a special pooling layer to account for the hemodynamic response function (HRF). This layer takes as input a fixed number of tokens $k = 20$, which are the last k tokens in the sequence, and learns a weight to apply to each token (note these weights are not element-wise). The bottleneck representations associated with the tokens are then combined according to these weights, and the resulting representation is used to predict the fMRI values using a linear layer. For fMRI prediction, we have a weight vector associated with the prediction of each individual voxel for each participant.

5.3.3 Computing task similarities

After our model has been trained, we use it to relate tasks to each other. To accomplish this, we use the weights that map the bottleneck representation (or the appropriate pooled version of the bottleneck representation) to the output layer of the model. Note that we cannot use a metric on the output predictions over all tasks, because our model cannot make predictions for some tasks from the input examples of other tasks. For example, tasks such as semantic role labeling which require spans to be given in the input are undefined on the examples for eye-tracking. In this section we discuss how these similarities are computed.

Task similarity computed with covariance-informed cosine similarity. Many different metrics could be used to assess the similarity/distance between tasks, but a natural starting point is to consider the correlation between model predictions (we cannot compute correlation directly since span-level tasks are undefined for examples that do not identify spans). Consider an ordinary least squares regression, with input matrix $X \in \mathbb{R}^{N \times M}$ which corresponds to N samples of M scalar random variables and two output vectors $y \in \mathbb{R}^N$ and $z \in \mathbb{R}^N$ which correspond to N samples of two scalar random variables. Then, the weights which minimize the residuals in this regression are $\beta_y = (X^\top X)^{-1} X^\top y$ and $\beta_z = (X^\top X)^{-1} X^\top z$. Premultiplying the weights by X gives us the model predictions for dataset X, which is also the projection of y (or z) onto the column space of X. Letting $\hat{y} = X\beta_y$ and $\hat{z} = X\beta_z$. The correlation of $\hat{y}$ and $\hat{z}$ tells us how well one variable, after it has been projected onto the column-space of X, can be linearly predicted from the other after it has been projected onto the column-space of X. Assuming that X, y, and z have

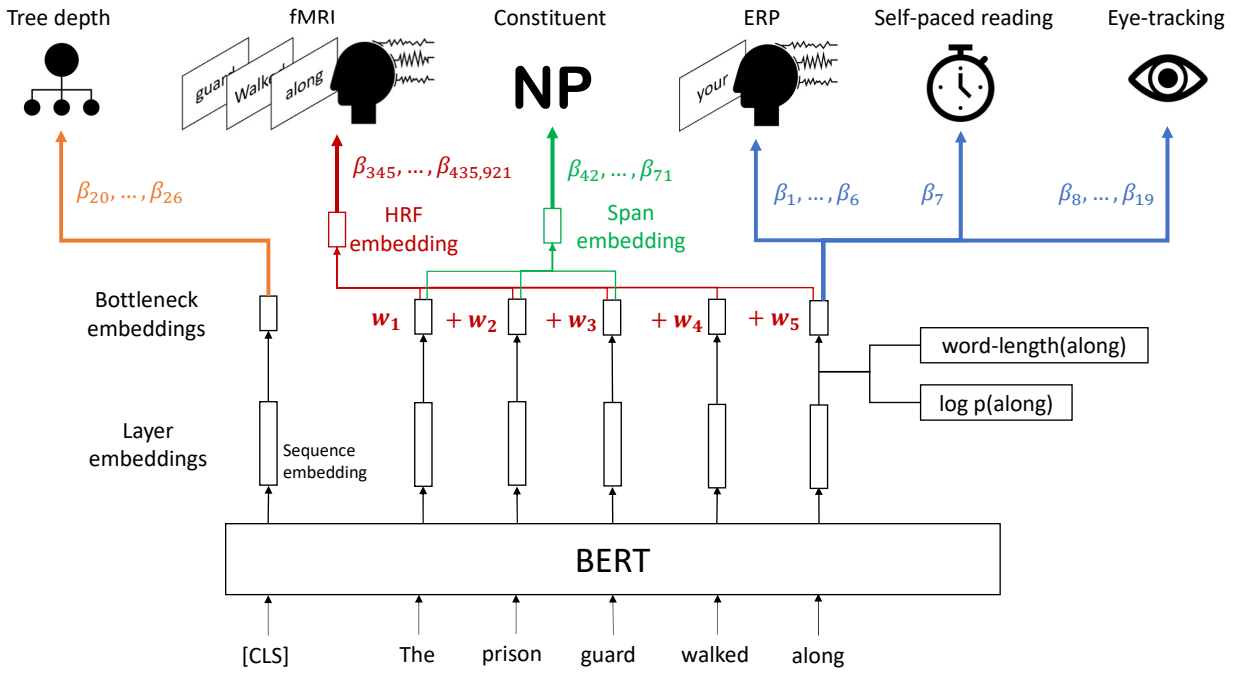


Figure 5.1: Schematic of model architecture showing the four architecturally different kinds of tasks we predict. **Shared encoder:** At the bottom of the model, in black, we first concatenate together the output embeddings from each layer of BERT along with the input embeddings to give a vector of $768 \times 13 = 9984$ components for each token, shown as "Layer embeddings" in the figure. To this, we concatenate word-length and word-frequency information. From this vector of 9986 components, we then compute a vector having 30 components using a linear layer followed by an elementwise Gaussian error linear unit (GELU) non-linearity (Hendrycks and Gimpel, 2016) and then layer normalization (Ba, Kiros, and Hinton, 2016). This part of the model can be thought of as learning 30 shared latent functions of the input sequence of text which support all of our tasks. **Word-level tasks:** At the right side of the model, in blue, word-level tasks are predicted directly from the bottleneck representation. Since there are six ERP tasks and twelve eye-tracking tasks there are six and twelve weight vectors associated with those icons respectively. **Sequence-level tasks:** At the left side of the model, in orange, sequence-level tasks are predicted directly from the bottleneck representation associated with the sequence ([CLS] token). Note that tree-depth is a multiclass task (a choice of seven depths). *One weight vector is associated with each class in our multiclass tasks*. **Span-level tasks:** The third task from the left, in green, is constituent prediction and shows how span-level tasks are predicted. The span of text we are trying to label (in this figure, *The prison guard*) is given to the model in the input. We combine the bottleneck embeddings associated with each token within the span by using a component-wise max pooling operation over those embeddings to create a single embedding for the span. From this embedding, we make a prediction for the task. Again note that constituent prediction is a multiclass task, so there is a weight vector associated with each class for this task. **fMRI:** For fMRI tasks (second from left), we learn a weight for each of the last 20 tokens in an input sequence (subscripted w in the figure) and combine together the bottleneck embeddings according to those weights to produce a hemodynamic response function (HRF) embedding. The weights are meant to allow the model to learn how to combine tokens to approximate the hemodynamic response delay. From the HRF embedding a weight vector is associated with each voxel in each participant's fMRI recordings, for a total of roughly 450,000 weight vectors and voxel predictions.

been centered, this correlation can be written as:

$$\rho(\hat{y}, \hat{z}) = \rho(X\beta_y, X\beta_z) = \frac{\beta_y^\top X^\top X \beta_z}{\sqrt{\beta_y^\top X^\top X \beta_y}\sqrt{\beta_z^\top X^\top X \beta_z}}$$

Note that we can also consider the predictions of the model on a new set of data D, even though β_y and β_z were not fit on D:

$$\rho(D\beta_y, D\beta_z) = \frac{\beta_y^\top D^\top D \beta_z}{\sqrt{\beta_y^\top D^\top D \beta_y}\sqrt{\beta_z^\top D^\top D \beta_z}}$$

This leads to the definition of covariance-informed cosine similarity, which, for ordinary least squares, would give the correlation of the model predictions on the data from which the covariance matrix C is computed. For weight vectors β_t and β_s associated with tasks t and s, the covariance-informed cosine similarity is:

$$\cos_{cov}(\beta_t, \beta_s, C) \coloneqq \frac{\beta_t^\top C \beta_s}{\sqrt{\beta_t^\top C \beta_t}\sqrt{\beta_s^\top C \beta_s}}$$

In our deep model, the intuition behind this measure is complicated by span-identification, pooling, differences between loss functions across tasks, and the fact that the covariance matrix is not truly a constant across tasks. However, if we ignore those effects, we can roughly think of this measure as the correlation of the model's predictions for tasks t and s on the validation data when C is the covariance matrix of the model's last shared layer over all of the validation data.

Covariance-informed cosine is robust to correlated input components. We have defined covariance-informed cosine similarity above as:

$$\cos_{cov}(\beta_t, \beta_s, C) \coloneqq \frac{\beta_t^\top C \beta_s}{\sqrt{\beta_t^\top C \beta_t}\sqrt{\beta_s^\top C \beta_s}}$$

Meanwhile, the standard cosine similarity of β_t and β_s can be written:

$$\cos(\beta_t, \beta_s) = \frac{\beta_t^\top \beta_s}{\sqrt{\beta_t^\top \beta_t}\sqrt{\beta_s^\top \beta_s}}$$

The change between $\cos_{cov}(\beta_t, \beta_s, C)$ and $\cos(\beta_t, \beta_s)$ is determined by the nature of the covariance matrix C. Consider the singular value decomposition $C = VSV^\top$ of C (note that the left and right singular vectors are equal because C is symmetric). Then, we can think of $\beta_t^\top V = (V^\top \beta_t)^\top$ as a rotation of β_t, and likewise $V^\top \beta_s$ as the same rotation of β_s, which does not change the angle between them. The change between the standard cosine similarity and the covariance-informed cosine is therefore determined by the singular values (the eigenvalues of the covariance matrix), which rescale the components of the weight vectors after the rotations. If we consider that $\cos(\beta_t, \beta_s)$ measures the angle between the two vectors $\frac{\beta_t}{||\beta_t||}$ and $\frac{\beta_s}{||\beta_s||}$ on a spheroid, then $\cos_{cov}(\beta_t, \beta_s, C)$ is measuring the angle between (the rotated) β_t and β_s on an ellipsoid where the lengths of the principle axes of the ellipsoid are determined by the eigenvalues of the covariance matrix C. By the same singular value decomposition argument, the covariance multiplication rotates the vectors onto an orthogonal basis before they are compared to each other. This makes the

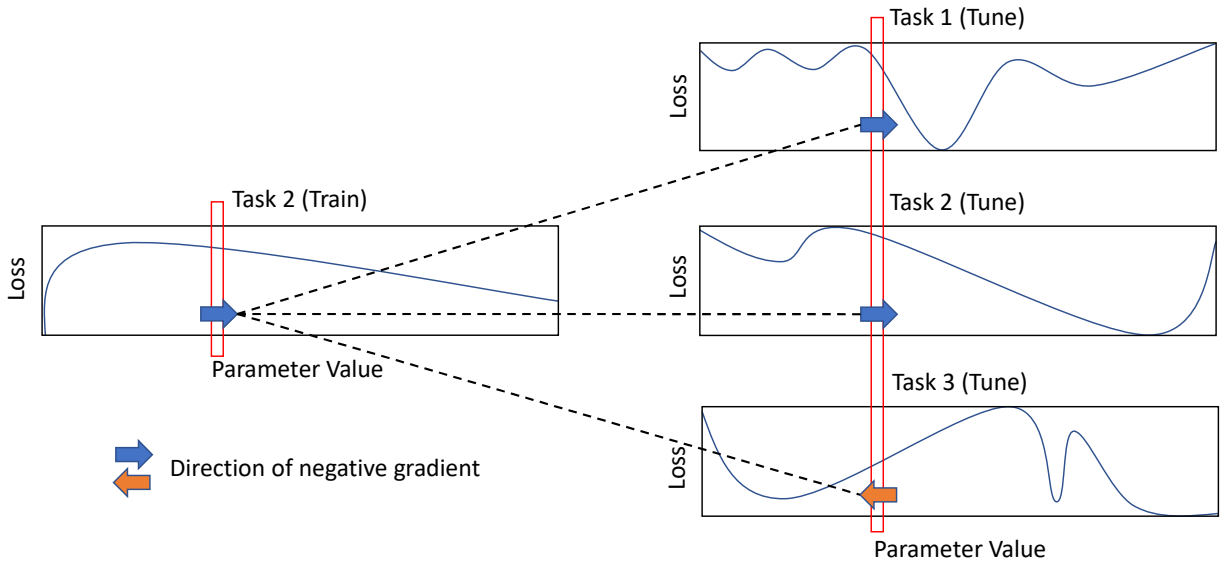


Figure 5.2: Illustration of how the MultiDDS algorithm balances tasks. Here we use a model with a single parameter to illustrate the algorithm. At left, the plot shows the loss on Task 2 on our training set for each value of this parameter, and at right the plots show the losses on Tasks 1, 2, and 3 for each value of this parameter on the reserved task-sample-rate tuning set. The intuition behind the MultiDDS algorithm is that we want to learn from a task more frequently if it tends to move the shared model parameters in a direction which improves the loss for all tasks on unseen data. In this illustration, the direction of the gradient on Task 2 on the training data is compared to the direction of the gradient on all tasks on the tuning data. When the direction is similar on average, Task 2 is moving the parameter in a direction that is good for all tasks, so the frequency with which Task 2 is sampled is increased. If the direction is dissimilar on average the frequency with which Task 2 is sampled is decreased. In the original formulation, each time the task sampling distribution is updated, the similarities in gradient direction are computed across all tasks, requiring a number of forward and backward passes through the model which is quadratic in the number of tasks. In our setting with roughly 80 tasks, this is intractable, so we modify the algorithm to sample the similarities in direction (see text).

covariance-informed cosine robust to redundant dimensions in the input. If we were to make a copy of one component of the input and concatenate this onto the existing input, then even if the model puts weight on the original input component when predicting one task and on the duplicated input component when predicting a second task, the orthogonalization will cause these two weight components to "collapse" together in the similarity measure.

5.3.4 Task sampling

One of the issues which needs to be addressed in multitask training is how to balance the different tasks against each other. Different tasks may have different numbers of examples, different difficulties, different signal-to-noise ratios, and potentially different priorities in a given application. When tasks have drastically different numbers of examples, then a model may overfit to a task with fewer examples before it is able to fit a task with more examples. When tasks have different priorities for an application, we'd like to prioritize finding a better fit for the more important tasks over finding a better fit for the less important ones. One approach to address the task balancing problem is to reweight the loss functions on each task to change

how much influence a task has on the model which is ultimately trained. A second approach is to change how frequently the model trains on a particular task by changing how training batches are selected. We use both approaches in this work, but here we focus on how we select batches of examples for training. To train our model, we use a modification of the MultiDDS algorithm proposed by Wang, Tsvetkov, and Neubig (2020). The intuition behind the algorithm is to learn from the tasks which push the shared model parameters in a direction which reduces the loss on all of the tasks on unseen data. To evaluate this, part of the data for each task is reserved to adjust how frequently each task is sampled during learning. We refer to this as the "tuning" set. In the original formulation of MultiDDS, after every M steps of gradient descent, for each task t, the gradient is computed with respect to the shared model parameters on the loss for task t on the training data, and a temporary step is taken in the negative direction of that gradient. Then, on the tuning data, the gradient is computed with respect to the shared model parameters using the temporarily modified parameter values. When the gradient direction for task t on the training data is on average similar to the gradient directions for all tasks on the tuning data (according to cosine similarity), the frequency with which task t is sampled is increased. If the direction is on average not similar, the frequency with which task t is sampled is decreased. See figure 5.2. In the original formulation, the number of forward and backward passes through the model required to compute the gradient direction similarity is quadratic in the number of tasks. In our setting with roughly 80 tasks, this is too expensive. Therefore, we modify the algorithm so that rather than waiting for M gradient descent steps and then computing the similarities over all pairs of tasks, after each gradient descent step on the training data, we sample one task on the tuning data and compute the similarity in the gradient directions between just those two tasks. We keep an exponential moving average of the gradient direction similarities between pairs of tasks, and after M gradient descent steps, we use these exponential moving averages to see how similar the direction of the gradient for task t on the training data tends to be to the directions for all tasks on the tuning data. This requires only twice the number of forward and backward passes through the model as in the standard gradient descent steps and avoids the quadratic complexity in the number of tasks. We also modify the standard MultiDDS algorithm by using a weighted average of the similarities between the gradient for a task t on the training data and the gradients for all tasks on the tuning data, so that we can express that it is more important for the direction of the gradient of task t to agree with the direction of the gradient for fMRI tasks (for example) than with other tasks (i.e. that we care more about reducing the loss on fMRI tasks). For a more complete description of the algorithm, see appendix C.

Restricting to predictable tasks and quantifying sampling error in task similarities. Task similarities are only meaningful if both tasks being compared are predictable (otherwise the function learned by the model for predicting a task is essentially random). We therefore only include a task in our similarity computation if the model can predict it better than a null baseline. For regression tasks, this baseline is that the correlation between task predictions and the actual values must be statistically significantly greater than 0, and for classification tasks this baseline is that the accuracy must be statistically significantly better than predicting the mode on the validation data. We compute p-values for this criterion using a one-tailed t-test. For fMRI, we use the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) within each participant to control for multiple comparisons over voxels, and we also apply Benjamini-Yekutieli over the ERP components, and separately over all of the eye-tracking data, all at a 0.01 false discovery rate. We use uncorrected p-values for all other tasks, applying a 0.01 significance threshold.

In order to quantify the sampling error in our task similarities due to the random initialization of the weights and the variation in how we sample the data, we want to estimate a mean and standard error for our similarities over model initializations. To accomplish this we use bootstrapping. Within each bootstrap resampling, our first step is to identify which prediction tasks are predictable as just described, and to filter

out the tasks which are not predictable. Then, because we have so many fMRI voxel prediction tasks (nine participants with roughly 50000 voxels each for a total of about 450000 voxel predictions), we split the fMRI tasks off from the other tasks and compute the similarity in two phases. In the first phase, we compute the pairwise similarity between all pairs of tasks in the set of significantly predictable non-fMRI tasks, giving a tensor of size $R \times T \times T$ where T is the number of non-fMRI significantly predictable tasks and $R = 100$ is the number of model initializations in the resampling. We next compute the similarity between weight vectors associated with significantly predictable fMRI voxels and the T significant non-fMRI tasks. We take the mean over model initializations (within the bootstrap resampling), to give one similarity vector for every significantly predictable task. The resulting matrix of size $(T + F) \times T$, with F the number of significantly predictable voxels, gives the mean similarity between all significantly predictable tasks (both non-fMRI and fMRI), and the significantly predictable non-fMRI tasks, and this is the result of interest for each bootstrap sample. Once we have $B = 100$ bootstrap samples of the similarity scores, we can estimate a mean and standard error on the similarity profiles for each task. In this final estimate, we include only those tasks which are significantly predictable across all of our bootstrap samples.

Removal of infrequently occurring classes. After computing task similarities, we observe that for several cases of multi-class predictions, a large majority of the representations for individual classes are nearly identical — see for example figure D.11. A common theme which emerges from the data is that representations of infrequently occurring labels are all nearly identical. We presume that the model learns to represent all the infrequent classes together as a pseudo-class of "infrequent" labels, and since these are uninformative, we apply a filter to remove them from most of our results. We filter out any classes where the label does not appear on at least 1% of examples in the training data, a threshold which was low enough to be conservative but still remove most of the uninformative similarities. For semantic role labeling, this preserves $8/66$ classes. For part of speech $21/48$ classes, for constituent labeling $10/30$ classes, for dependency roles $21/49$ classes, for named entity recognition $11/18$ classes, and for SemEval $17/19$ classes. We also apply this filter to the proto-role classes, where although the labels are binary and each example can have multiple labels, we filter out any class which is not positive on at least 1% of examples. This leaves $16/18$ for the pr1 set, and $13/20$ for the pr2 set. All other binary classification tasks are well balanced.

fMRI summaries. Because we have too many fMRI voxel predictions to easily look at each one-by-one, we use the following procedure to compute something like a weighted average over voxels within eight regions of interest. We first filter the voxels to those where the correlation between the predicted and actual value of the voxels is significantly greater than 0 at a 0.01 level according to a one-tailed, one-sample t-test corrected for multiple comparisons using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001). Then, we separate the voxels into regions of interest using an updated version of the regions that are functionally defined in (Fedorenko, Hsieh, et al., 2010). The regions are transformed to an individual participant's brain space using PyCortex (Gao et al., 2015). For each voxel, we consider only similarities to non-fMRI classes/tasks that are significantly predictable. Additionally, only classes/tasks that appear in at least 1% of examples are considered. Similarities which are not more than 2 standard errors away from 0 according to our bootstrapping estimate are not considered. Among the surviving similarities for a voxel, we keep only the 20 similarities with the largest magnitude and mask out all others. Then to create the summary for one region of interest of one participant, we take a sum weighted by the correlation between the predicted and actual value for a voxel, and normalized by the maximum surviving weights within a class. Let s_i be the ith component of the summary for one region for one participant. This ith component corresponds to the ith non-fMRI class/task after filtering for significant predictability and $> 1\%$ frequency

in the examples as described above. Let r_j be the correlation between the predicted and actual values for the jth voxel of J total voxels which survive our filtering within that participant's region, and $c_{j,i}$ be the similarity between the prediction of voxel j and task i according to our similarity measure. Let $n_{j,i} = 1$ if i is among the 20 largest values of $|c_{j,i}|$ for a fixed j and $n_{j,i} = 0$ otherwise. Then we can write the equation for s_i as:

$$s_i = \frac{\sum_{j=1}^{J} r_j \cdot c_{j,i} \cdot n_{j,i}}{\max_k \sum_{j=1}^{J} r_j \cdot n_{j,k}}$$

5.4 Results

5.4.1 Validity of similarities

Task similarities recover equivalent tasks and intuitively similar tasks. The task similarities we compute using our method recover the relationships between tasks that we would intuitively expect. In table 5.3, we show the eighty highest magnitude similarities among all pairs of tasks after excluding within-corpus tasks, and excluding any pair where both tasks are eye-tracking (if we do not exclude intra-corpus tasks, the most similar tasks are dominated by highly correlated semantic proro-role properties and eye-tracking properties; after these exclusions we have about $25K$ possible pairs of tasks). We first note that across semantics and syntax, nearly equivalent tasks on different datasets have very high similarity. Specific examples include:

- pos.CC (part of speech, coordinating conjunction, e.g. ***and***, and ***or***) and dp.cc.sp0 (dependency role, dependent of the coordinating conjunction relation, i.e. a coordinating conjunction). These are equivalent by definition, and the mean similarity over 100 model initializations is 0.95, a strong statement about the similarity of these tasks under our model.
- pos.DT (a determiner, e.g. ***the***, ***a***) and dp.det.sp0 (also a determiner) score 0.95.
- pos.IN (a preposition that precedes either a prepositional phrase, e.g. ***in*** *the basement* or a subordinate clause, e.g. ***whenever*** *he went to the store*) and dp.case.sp0 (a preposition in a prepositional phrase — e.g. ***in*** *the basement*, or the 's in a possessive clitic — e.g. *Maria****'s*** book). These two have a high frequency intersection in their definition — namely prepositions in prepositional phrases, and score 0.95 in task similarity.
- pos.PRP$ (a possessive personal pronoun, e.g. ***my***, ***your***, ***her***), and dp.nmod:poss.sp0 (the possessor in the head position of a possessive, e.g. ***Maria****'s book*). These are not equivalent, but have very similar semantics, and score 0.89.
- pos.RB (an adverb, e.g. ***quickly*** *wrote*) and dp.advmod.sp0 (the adverb in an adverbial modifier, e.g. ***quickly*** *modified text*). These are nearly equivalent, and score 0.87.
- pos.MD (a modal verb, e.g. ***shall***, ***could***, ***will***) and sr.ARGM-MOD.sp1 (a modal verb). These are equivalent and score 0.87.
- pos.MD (a modal verb, e.g. ***shall***, ***could***, ***will***) and dp.aux.sp0 (an auxiliary verb, including modal verbs but also including periphrastic tense — an inflection expressed through multiple words, e.g. ***have*** *gone*, ***has been*** *taken*). These are not quite equivalent, but have very similar semantics, and score 0.84.
- pos.CD (a cardinal number, e.g. ***5***) and dp.nummod.sp0 (a number which modifies as a noun, e.g. ***5*** *cars*). These are very close to equivalent, and score 0.86.

- pos.JJ (an adjective, e.g. ***blue***) and dp.amod.sp0 (an adjective which modifies a noun phrase, e.g. ***blue*** *potatoes*) are nearly equivalent and score 0.83.
- pos., (a comma) and dp.punct.sp0 (punctuation). The first is a subset of the second, and these score 0.78.
- c.ADVP (a phrase which functions as an adverb, e.g. ***in the morning***) and dp.advmod.sp0 (the adverb in an adverbial modifier, e.g. ***quickly*** *modified text*). These both function as adverbs, and score 0.77.

Along with these tasks that are nearly equivalent by definition, our method also finds similarities that are very intuitive. In table 5.3, we see many examples of se.Instr-Agn(e2,e1).sp0 and se.Instr-Agn(e1,e2).sp1 — both the agent in the SemEval Instrument-Agency relation. Similarly, we see many examples of the se.Pdct-Pdcr(e2,e1).sp0 and se.Pdct-Pdcr(e1,e2).sp1 — both the producer in the Product-Producer relation. The agent and producer in those SemEval relations are highly similar to the proto-role awareness property (pr1.awareness.sp1/pr2.awareness.sp1) as well as volition (pr1.volition.sp1/pr2.voltion.sp1), instigation (pr1.instigation.sp1), and sentient (pr1.sentient.sp1/pr2.sentient.sp1). The similarities are highly intuitive since an agent or producer in the SemEval relations should have the agent-like, animacy aspects described by those similar proto-role properties. Likewise, we see that the producer and agent are anti-similar to the manipulated-by-another property (pr1.manip_by.sp1), a property which is strongly indicative of a proto-patient, and which intuitively is a mismatch to the producer or agent in these SemEval relations. A different pairing that our method considers similar is sr.ARG0.sp1 (the agent-like argument in a sentence), and dp.nsubj.sp0 (the subject in a clause). These are not the same, but they are highly correlated in regular language use and intuitively similar or indeed confusable. These results are all encouraging. Additionally, there are not really any cases among these high scoring similarities that are difficult to understand.

Symmetry of SemEval tasks. A second line of evidence for model validity is the symmetry we see between similarities of SemEval tasks. Figure 5.3 shows a similarity matrix of how the SemEval tasks compare to each other, with the heatmap showing similarity on a scale from -1 to 1. The SemEval tasks are interesting because they include directionality (e.g. the label for Cause-Effect(e1,e2) is true when the argument in the first span is a cause of the argument in the second span, while the label Cause-Effect(e2,e1) is true when the argument in the second span is a cause of the argument in the first span). For that reason, we show spans 0 and 1 integrated in a single similarity matrix, which is arranged such that for each relation we have in order: relation(span0, span1).span0, relation(span0, span1).span1, relation(span1, span0).span0, relation(span1, span0).span1. Ideally then, for each block of sixteen squares along the diagonal, the four corners should be highly similar to each other, the center four squares should be highly similar to each other, and the eight non-corner edge squares should be less similar to each other than the center and corners (these would not necessarily be expected to be dissimilar, since they capture information which is relevant to the relation, but with reversed direction). We see in figure 5.3 that this pattern largely holds. One exception is in the overlap between the Entity-Origin and Entity-Destination relation, which is a sensible area for model confusion due both to the infrequency of the e2,e1 direction of the Entity-Destination direction, and the semantic overlap between the Entity-Origin and Entity-Destination relations. Several off-block-diagonal similarities are also apparent. Two off-diagonals show the directionality pattern we expect (similar center and corners): these are Instrument-Agency:Product-Producer and a weaker Cause-Effect:Message-Topic similarity. Other off diagonal similarities are stronger in one direction than in the other, for example Instrument-Agency:Member-Collection is stronger in the Agency→Instrument direction than the Instrument→Agency direction. This may reflect the infrequency of the e1,e2 direction of the Member-Collection relation, but it could also be from other noise.

Task 1	Task 2	Sim.	Task 1	Task 2	Sim.
pos.CC	dp.cc.sp0	0.95	pos.VBZ	dp.cop.sp0	0.80
pos.DT	dp.det.sp0	0.95	pos.PRP	dp.nsubj.sp0	0.80
pos.IN	dp.case.sp0	0.93	sr.ARG0.sp1	pr1.volition.sp1	0.79
se.Instr-Agn(e2,e1).sp0	pr1.awareness.sp1	0.92	se.Instr-Agn(e1,e2).sp1	pr2.awareness.sp1	0.79
se.Instr-Agn(e2,e1).sp0	pr1.volition.sp1	0.92	se.Instr-Agn(e2,e1).sp0	pr1.sentient.sp1	0.79
se.Pdct-Pdcr(e2,e1).sp0	pr1.volition.sp1	0.91	sr.ARG0.sp1	pr1.existed_before.sp1	0.79
se.Pdct-Pdcr(e2,e1).sp0	pr1.awareness.sp1	0.91	se.Instr-Agn(e1,e2).sp1	pr1.instigation.sp1	0.79
se.Instr-Agn(e2,e1).sp0	pr2.volition.sp1	0.91	ne.PERSON	pr1.sentient.sp1	0.79
se.Pdct-Pdcr(e1,e2).sp1	pr1.awareness.sp1	0.91	pos.,	dp.punct.sp0	0.78
se.Instr-Agn(e2,e1).sp0	pr2.awareness.sp1	0.90	sr.ARG0.sp1	pr2.awareness.sp1	0.78
se.Pdct-Pdcr(e1,e2).sp1	pr1.volition.sp1	0.90	sr.ARG0.sp1	pr1.instigation.sp1	0.78
se.Pdct-Pdcr(e2,e1).sp0	pr2.volition.sp1	0.89	sr.ARG0.sp1	pr1.manip_by.sp1	-0.78
pos.PRP$	dp.nmod:poss.sp0	0.89	se.Instr-Agn(e1,e2).sp1	pr2.sentient.sp1	0.78
se.Instr-Agn(e2,e1).sp0	pr1.manip_by.sp1	-0.88	ne.PERCENT	pr1.existed_during.sp1	-0.77
se.Instr-Agn(e2,e1).sp0	pr1.instigation.sp1	0.88	c.ADVP	dp.advmod.sp0	0.77
se.Pdct-Pdcr(e1,e2).sp1	pr2.volition.sp1	0.88	sr.ARG0.sp1	pr2.volition.sp1	0.76
se.Pdct-Pdcr(e2,e1).sp0	pr2.awareness.sp1	0.88	pos.VBG	dp.advcl.sp0	0.76
se.Pdct-Pdcr(e1,e2).sp1	pr2.awareness.sp1	0.88	se.Instr-Agn(e1,e2).sp1	pr1.sentient.sp1	0.76
pos.RB	dp.advmod.sp0	0.87	se.Instr-Agn(e1,e2).sp1	pr1.manip_by.sp1	-0.75
pos.MD	sr.ARGM-MOD.sp1	0.87	c.NP	sr.ARG1.sp1	0.75
sr.ARGM-MOD.sp1	dp.aux.sp0	0.87	sr.ARGM-TMP.sp1	ne.DATE	0.75
se.Pdct-Pdcr(e2,e1).sp0	pr1.manip_by.sp1	-0.87	c.WHNP	sr.R-ARG0.sp1	0.75
se.Instr-Agn(e2,e1).sp0	pr2.sentient.sp1	0.87	sr.ARG0.sp1	pr1.existed_after.sp1	0.75
se.Pdct-Pdcr(e2,e1).sp0	pr1.instigation.sp1	0.87	c.NML	dp.compound.sp0	0.74
pos.CD	dp.nummod.sp0	0.86	pos.VBN	dp.advcl.sp0	0.74
pos.MD	dp.aux.sp0	0.84	dp.nsubj.sp0	pr1.existed_before.sp1	0.74
sr.ARG0.sp1	dp.nsubj.sp0	0.84	sr.ARG0.sp1	pr2.sentient.sp1	0.74
se.Pdct-Pdcr(e1,e2).sp1	pr1.instigation.sp1	0.84	dp.nsubj.sp0	pr1.existed_during.sp1	0.74
se.Pdct-Pdcr(e1,e2).sp1	pr1.manip_by.sp1	-0.84	pos.VBP	dp.aux.sp0	0.73
pos.TO	dp.mark.sp0	0.84	sr.ARG0.sp1	se.Pdct-Pdcr(e1,e2).sp1	0.72
se.Pdct-Pdcr(e1,e2).sp1	pr2.sentient.sp1	0.84	se.Mem-Coll(e2,e1).sp1	pr1.sentient.sp1	0.72
pos.RB	c.ADVP	0.83	pos.NNP	pr1.existed_before.sp1	0.72
pos.JJ	dp.amod.sp0	0.83	pos.NNP	pr1.existed_during.sp1	0.72
se.Pdct-Pdcr(e2,e1).sp0	pr2.sentient.sp1	0.83	sr.ARG0.sp1	se.Pdct-Pdcr(e2,e1).sp0	0.72
se.Instr-Agn(e1,e2).sp1	pr1.awareness.sp1	0.82	se.Cause-Effect(e1,e2).sp1	pr1.existed_before.sp1	-0.72
se.Instr-Agn(e1,e2).sp1	pr1.volition.sp1	0.82	se.Pdct-Pdcr(e1,e2).sp1	pr1.sentient.sp1	0.71
sr.ARG0.sp1	pr1.existed_during.sp1	0.81	se.Cause-Effect(e2,e1).sp0	pr1.existed_before.sp1	-0.71
se.Instr-Agn(e1,e2).sp1	pr2.volition.sp1	0.81	se.Pdct-Pdcr(e2,e1).sp0	pr1.sentient.sp1	0.71
sr.ARG0.sp1	pr1.awareness.sp1	0.81	pos.NN	dp.xcomp.sp1	-0.71
se.Instr-Agn(e1,e2).sp1	pr2.volition.sp1	0.81	ne.ORG	pr1.existed_during.sp1	0.71

Table 5.3: First eighty task pairs sorted in descending order of the magnitude of their similarity with intra-corpus pairs removed, and all eye-tracking vs. eye-tracking pairs removed.

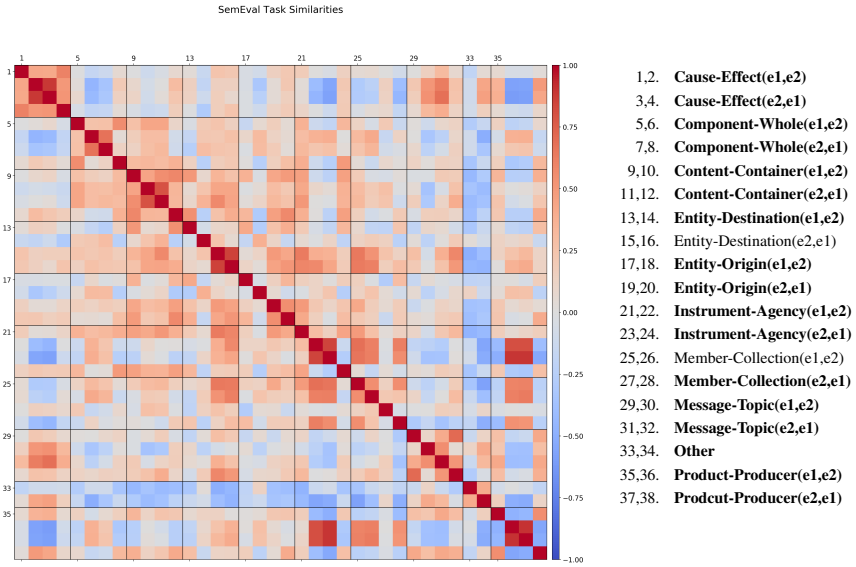


Figure 5.3: Similarities of SemEval classes to each other. Span 0 (which corresponds to e1 in a relation) and span 1 (which corresponds to e2 in a relation) are rendered in alternating fashion in this similarity matrix. Color in the matrix indicates similarity on a scale from -1 to 1. Grid lines separate relations from each other. The enumerated list on the right shows the order in which the classes are rendered. Bold indicates classes which appear in at least 1% of examples (here we have not filtered out classes which appear in fewer than 1% of examples in order to better see the symmetry). See section 5.4.1 for discussion.

5.4.2 Using task similarity to better understand model mechanisms

Having established that our method rates equivalent or intuitively similar tasks as similar even across datasets, we turn to how we can use similarity profiles (the 20 tasks of the possible 237 with the highest magnitude similarities to a given task) to get an understanding of the model's mechanisms of prediction. As a case study, we examine semantic role labels and their relationship to semantic proto-role properties (decompositional semantic properties) from prior work, and then turn to what our similarity profiles tell us about the prediction of ARG0 in the semantic role labeling task.

Semantic role labeling. The semantic role labeling task, described in Palmer, Gildea, and Kingsbury (2005) and later extended to the OntoNotes v5.0 dateset (Weischedel et al., 2011) (we use the version from Tenney et al. (2019)) uses a verb-specific labeling scheme to identify the roles that arguments play with respect to a verb. The labels — ARG0, ARG1, ARG2, etc. — are only defined with respect to the verb that is being labeled. For example, for the verb *expect*, ARG0 is defined as the *expecter* and ARG1 is defined as the *thing expected*. For the verb *give*, ARG0 is defined as the *giver*, ARG1 as the *thing given*, and ARG2 as the *beneficiary*. ARG0 is mostly agent-like, ARG1 is mostly patient-like, and ARG2 mostly acts as a non-core argument such as a benefactive, instrument or attribute. In the rest of this discussion, we only concern ourselves with ARG0, which we can think of as the agent of a sentence. In the sentence *John gave the book to Mary*, John acts as the agent and should be given the label ARG0.

Semantic proto-roles. While the standard semantic role labeling guidelines embrace the concept of a large number of roles for the arguments in a sentence (i.e. the labels are defined differently for every verb), decompositional semantics (Dowty, 1991) proposes that instead of creating roles for every verb

or trying to create an inventory of roles that works across all verbs, a more sensible approach is to break down monolithic concepts like the "agent" and "patient" of a proposition into a small set of properties. Collections of properties form prototypical roles, so that the proto-agent generally has the property of awareness of the action, volition in performing an action and so on. Dowty (1991) suggests that the distribution of the combinations of these properties in real language usage has a long tail, so that there are a few frequently occurring prototypical roles like the proto-agent and proto-patient, but there are also many other variations which may, e.g., lack some of the proto-agent properties in a particular context. To empirically test the theory of semantic decomposition, Reisinger et al. (2015) used crowdsourcing to label part of the original PropBank (Palmer, Gildea, and Kingsbury, 2005) with properties inspired by the theory of Dowty (1991). We have incorporated that dataset, "semantic proto-roles 1", as a binary set of tasks, abbreviated pr1 in our work. Subsequently, part of the English Web Treebank was also labeled with a modified set of properties (White, Rastogi, et al., 2017), and we use this set of binary tasks as "semantic proto-roles 2", abbreviated pr2 in our work. We use the version of these tasks from Tenney et al. (2019).

ARG0 and agent-like decompositional semantic properties. White, Rawlins, and Van Durme (2017) examines the relationship between the decompositional semantic properties in the semantic proto-role labeling tasks and semantic role labels such as ARG0 and ARG1. White, Rawlins, and Van Durme (2017) refer to ARG0, ARG1, etc. as predicate-specific roles since they change definitions for each verb, and point out that predicate-general roles (roles not specific to a predicate, such as a proto-agent role) can help a model learn to predict these predicate-specific roles. The authors use a generative model which learns latent predicate-general roles as an intermediary in prediction of semantic role labels such as ARG0. In figure 5.5, we show a figure from White, Rawlins, and Van Durme (2017) which has two heatmaps for two of their generative models — a 2-role model in the left heatmap and a 6-role model in the right headmap. Each column in the heatmaps shows one latent proto-role, and the color in each row indicates how likely a decompositional semantic property is for the latent role associated with each column (black indicates a higher likelihood and red indicates a lower likelihood). By looking at how their model uses the latent roles to make predictions of the subject, object, and oblique in an input sentence as well as the semantic role labels ARG0-ARG5, the authors conclude that the leftmost column in the 2-role model and the leftmost 2 columns in the 6-role model match the proto-agent role described by Dowty (1991). We note that the 8 properties at the top portion of the heatmaps have strong likelihood of appearing on a proto-agent, and we can think of these as agent-like properties. The five columns in the heatmap not identified as proto-agent columns are all patient/non-agent proto-roles. We note the strong tendency of these roles to have the manipulated-by-another property (4 of 5 have high likelihood of this property) and the predicate-changed-argument property (4 of 5 have medium likelihood of this property). These two properties are thus very patient-like.

The weight vector associated with the argument span of the ARG0 class prediction reproduces results from prior work. In our model, semantic role labeling is a 2-span, multiclass task. The model is given one span which identifies the argument it must label, and a second span which identifies the verb with respect to which the argument is being labeled. For each of these spans, a span embedding is computed by taking the elementwise-maximum over the bottleneck embeddings associated with the token in each span. From the span embeddings, we have two weight vectors to predict each class probability output — in this case we concern ourselves only with the ARG0 class — one weight vector is associated with the verb-context span, and the other is associated with the argument span (see figure 5.4). The inner-products of these weight vectors with the input span-embeddings are added together to produce the prediction for a particular class. Thus, we have a similarity profile for the weight vector associated with the verb-context for

the ARG0 prediction ($\beta_{ARG0.V}$ in figure 5.4), and we have a similarity profile for the weight vector associated with the argument for the ARG0 prediction ($\beta_{ARG0.A}$ in figure 5.4). In figure 5.5, we see the similarity profile for $\beta_{ARG0.A}$ along with heatmaps from White, Rawlins, and Van Durme (2017) discussed in the previous paragraph. We see here that from a completely different method (task similarity), we get results which are very similar to the agent-like latent roles in the generative model developed by White, Rawlins, and Van Durme (2017), which we think is a very nice confirmation of the usefulness of similarity profiles for scientific exploration. Apart from the proto-role properties, the remaining tasks in the $\beta_{ARG0.A}$ profile are all also highly agent-like: dp.nsubj.span0 is the subject of a clause, se.Product-Producer(e1,e2).span1 and se.Product-Producer(e2,e1).span0 are both the producer in the Product-Producer SemEval relation, se.Instrument-Agency(e2,e1).span0 is the agent in the Instrument-Agency relation, PRP is a personal pronoun (e.g. he, she), NNP is a proper noun (e.g. Bob), and ner.ORG is a organization. Meanwhile, $\beta_{ARG0.A}$ is anti-similar to the effect in the Cause-Effect SemEval relation. Thus overall we see a very agent-like profile for $\beta_{ARG0.A}$.

ARG0, awareness, and the competition hypothesis. Figure 5.6 shows the similarity profiles for the weight vectors shown in figure 5.4 labeled $\beta_{ARG0.V}$ (the ARG0 verb-context profile), $\beta_{ARG0.A}$ (the ARG0 argument profile), $\beta_{AWR.V}$ (the pr1 awareness verb-context profile), and $\beta_{AWR.A}$ (the pr1 awareness argument profile). In contrast to the profile for $\beta_{ARG0.A}$, the profile for $\beta_{ARG0.V}$ is highly patient-like: it is strongly anti-similar to many of the agent-like decompositional semantic properties and similar to the manipulated-by-another property. It is also anti-similar to the personal pronoun label, the possessive personal pronoun label, the possessor in the nmod:poss dependency role relation, the coreferent in coreference prediction, and the producer in the SemEval Product-Producer relation — all highly agent-like. This patient-like profile for the weight vector between the verb context and the ARG0 prediction is puzzling at first glance. Why should part of the prediction of ARG0, which is essentially the agent of the sentence, be patient-like? We can get some insight by considering the problem the model needs to solve. Figure 5.4 shows the problem diagrammatically. The model needs to use the same input span embeddings for both the ARG0 prediction and the semantic proto-role awareness property prediction. When the argument being labeled is the agent of a sentence the model needs to assign high probability to both the ARG0 and the awareness predictions (because the argument is ARG0 and is aware the verb is happening), but if the argument being labeled is an animate argument other than the agent, the model needs to lower the probability for ARG0 but maintain a high probability for the awareness prediction. Unlike the ARG0 profiles, we see that both the $\beta_{AWR.V}$ and $\beta_{AWR.A}$ profiles for awareness are agent-like. Thus agent-like properties in the representation of the verb reinforce the awareness label but inhibit the ARG0 label. We note that ARG0 can only be assigned to a single argument in a clause while multiple arguments can be labeled as having the awareness property. One way that the model can solve the problem of correctly predicting both awarness and ARG0 from the same input span embeddings is to embed the agent-like properties of all of the arguments in a sentence into the verb embedding. If it does this, and makes $\beta_{ARG0.V} \approx -\beta_{ARG0.A}$, then the ARG0 prediction becomes something like predicting whether the current argument is the most agent-like argument in the sentence, and thus a good candidate for ARG0. We hypothesize that by embedding the agent-like properties of the arguments of the sentence into the verb, the model is thus using competition between those arguments to predict ARG0. This case study is a nice example of how we can use the task similarity profiles produced by our method can lead to illuminate the mechanisms that a model uses to make its predictions.

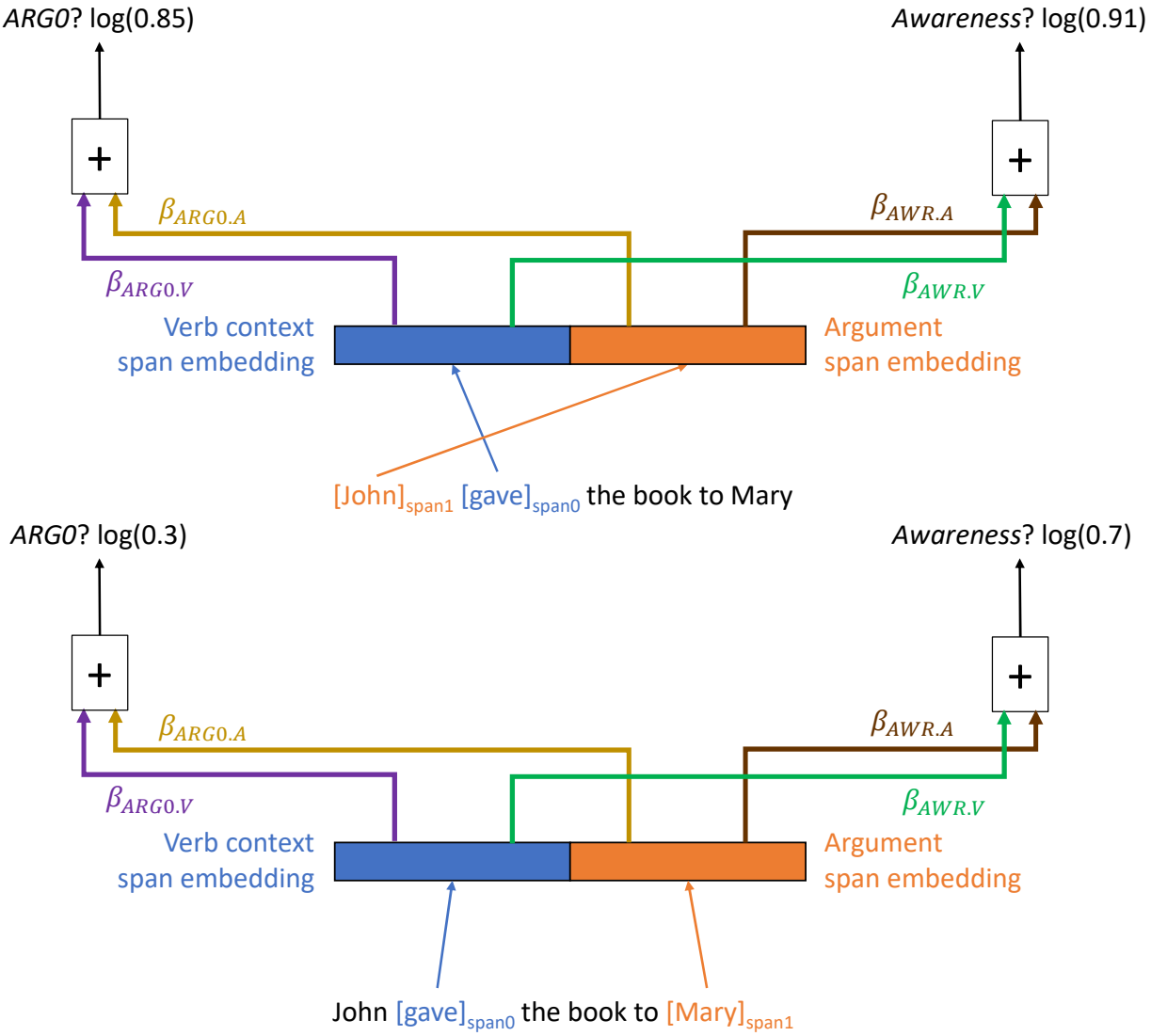


Figure 5.4: This figure illustrates how the ARG0 class prediction and Awareness property prediction are made by our model. In the top figure, the model needs to label *John* with respect to the verb *gave* in the sentence *John gave the book to Mary*. A span embedding is computed for the verb *gave* as described in section 4.3.2 and a separate span embedding is computed for the argument John. These embeddings are shared by the two tasks. The task outputs are differentiated only by the weight vectors which map from the span embeddings to the task outputs. When the model is making predictions for the argument *John*, since *John* is both the agent of the sentence (ARG0) and has awareness that the verb is happening, the output probabilities should be high for both tasks. When the model is making predictions for *Mary*, it must decrease the output probability for the ARG0 prediction (since *Mary* is not ARG0), but must keep the probability for awareness high (since *Mary* still has awareness that the verb is happening).

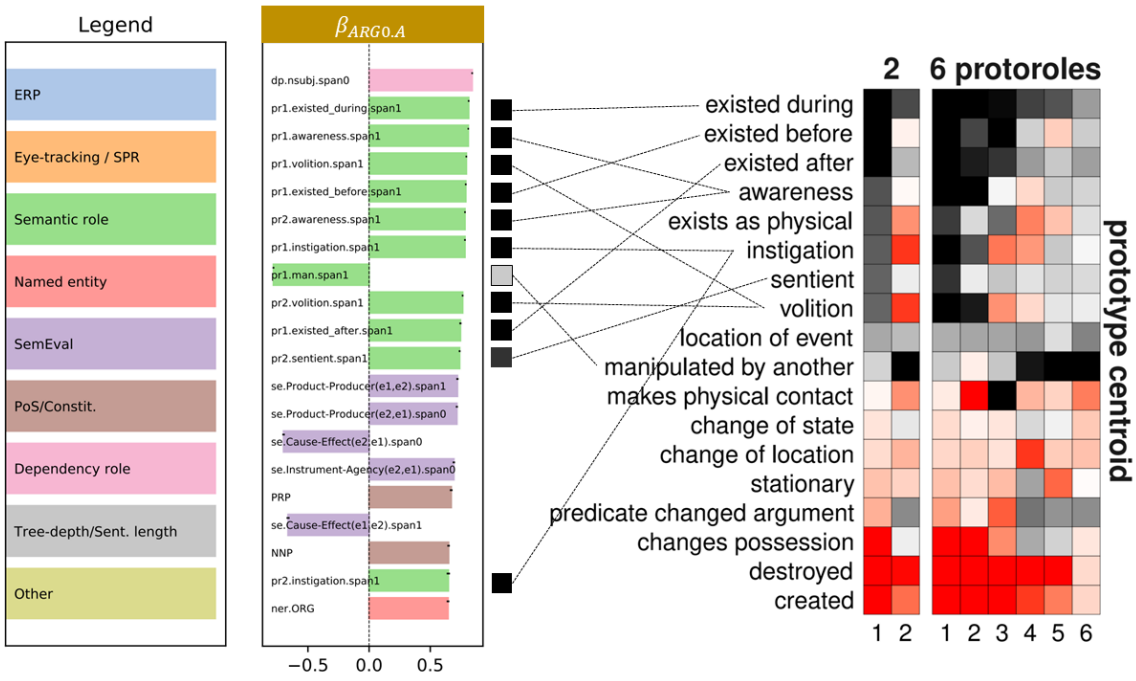


Figure 5.5: At left, we show the similarity profile for the weight vector associated with the argument span for the ARG0 class prediction, labeled $\beta_{ARG0.A}$ in figure 5.4. At right, figure 4 from (White, Rawlins, and Van Durme, 2017) shows how properties relate to the latent predicate-general roles which they use as a modeling intermediary for mapping tokens to predicate-specific roles (PropBank roles) for either 2 latent roles (left of subplot) or 6 latent roles (right of subplot). In the heatmap, black indicates higher likelihood and red indicates lower likelihood. The proto-agent role as described by Dowty (1991) matches the leftmost column of both subplots. In the right subplot with 6 roles, the proto-agent has been split into two subcategories (animate and inanimate) (White, Rawlins, and Van Durme, 2017). All other columns in the figure are described in White, Rawlins, and Van Durme (2017) as non-agent columns or perhaps different kinds of proto-patient. We observe that many of the weight vectors associated with the argument spans of semantic proto-role property predictions are similar to $\beta_{ARG0.A}$, and we connect these by dashed lines to the same properties in the heatmap from (White, Rawlins, and Van Durme, 2017) to emphasize the overlap. The figure also shows the colors from the first column of the 6-latent role heatmap next to each bar in the profile to emphasize that these are agent-like properties (and the anti-similarity to the patient-like manipulated-by property).

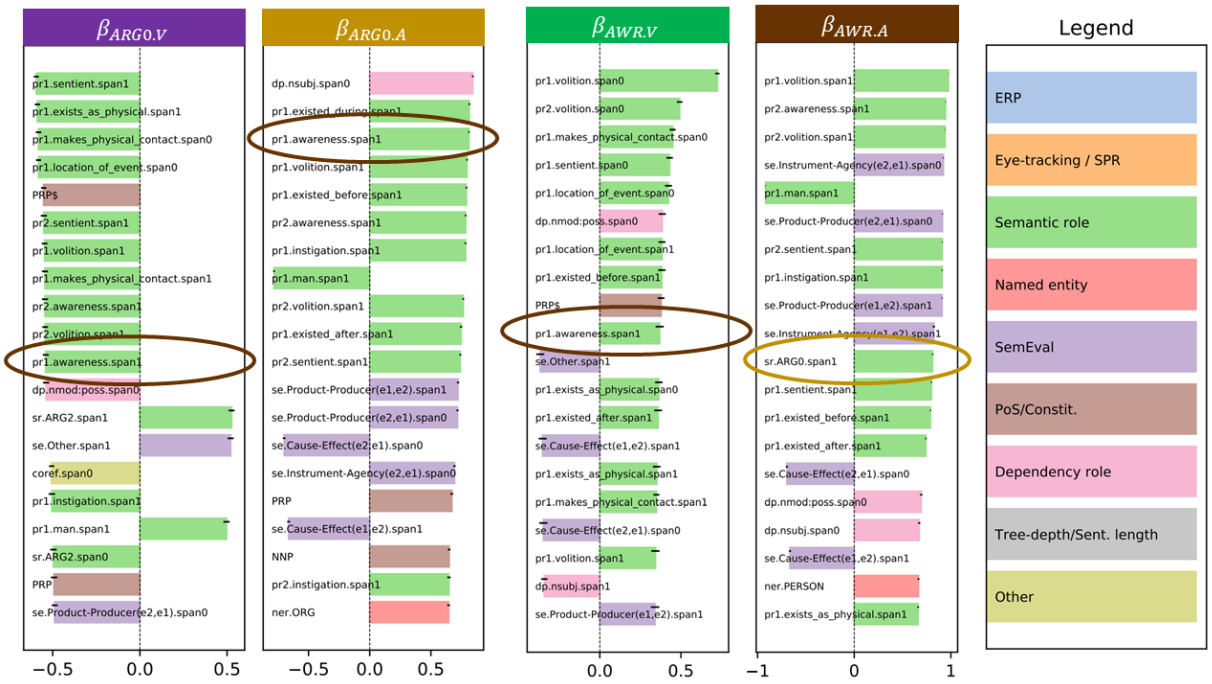


Figure 5.6: This figure shows the four similarity profiles of the weight vectors in figure 5.4 labeled $\beta_{ARG0.V}$ (the ARG0 verb-context profile), $\beta_{ARG0.A}$ (the ARG0 argument profile), $\beta_{AWR.V}$ (the pr1 awareness verb-context profile), and $\beta_{AWR.A}$ (the pr1 awareness argument profile). Note that both $\beta_{ARG0.A}$ and $\beta_{AWR.A}$ have agent-like profiles and are similar to each other (circled bars in profiles). $\beta_{AWR.V}$ is also agent-like, and similar to $\beta_{AWR.A}$ (circled bar), thus the two parts of the awareness prediction reinforce each other. Meanwhile, $\beta_{ARG0.V}$ has a patient-like profile and is anti-similar to many of the same tasks that $\beta_{ARG0.A}$ is similar to ($\beta_{AWR.A}$ is circled in both profiles to illustrate the point). Thus the verb-context part of the ARG0 prediction weakens the prediction coming from the $\beta_{ARG0.A}$ weight vector.

5.4.3 Other observations of NLP similarity profiles.

Along with the observation that the similarity profile of the argument span for ARG0 prediction resembles the results from White, Rawlins, and Van Durme (2017), and our hypothesis that ARG0 prediction uses competition between the arguments in a sentence by embedding the agent-like properties of those arguments into the representation of the verb, there are many other interesting properties of the NLP profiles which we discuss in detail in appendix D and summarize here. We note that throughout our analyses the similarity profiles are for the most part highly interpretable, and that interpretation of the profiles is often reliant on the semantic properties from the semantic proto-role tasks. We observe that there are other 2-span prediction tasks which exhibit symmetry similar to the symmetry we observe in ARG0 prediction which suggest that other predictions also use a similar competition between arguments. We also see that for many semantic task predictions, profiles are agent-like or patient-like in that they are similar or anti-similar to specific semantic proto-role properties. Semantics related to physicality, location, and temporal information are also important for the prediction of many of the semantic tasks. Syntactic tasks are, in general similar to other syntactic tasks and less similar to semantic tasks, with some exceptions. Thus the model appears to exhibit some separation of syntax and semantics in its latent functions shared functions. Finally, by examining sequence-level task similarity profiles, we see that sequence embeddings contain a surprising amount of information about the individual tokens in the sequence, which the model is able to make use of for sequence-level predictions.

5.4.4 Cognition-relevant similarity patterns

ERP component tasks are similar to proto-patient property tasks. Figure 5.7 shows the similarity profiles of the event related potential (ERP) component tasks. The ERP tasks are foremost similar to each other (and note that the similarities are all positive). The ERP components are functionally defined in this dataset as the average of a predefined set of EEG sensors during predefined time-windows (see figure 3.1 in chapter 3), and seen in this light it is sensible that they would be correlated and that their signs would therefore be aligned. Across all ERPs the similarity profiles seem to be a strong match to a proto-patient profile, and all have strong similarity to the weight vector associated with the argument span of the semantic role ARG2 prediction (ARG2 is a non-core argument to a verb such as a benefactive, instrument, attribute, etc., see figure D.3 for profile). The N400 and P600 are also similar to self-paced reading time and to go-past time (the summed duration of all fixations on a word and any preceding word up to the first fixation on a subsequent word, go past-time is a "late" measure of context complexity), while the LAN is similar to go-past time. The patient-like profile of the ERP components might be related to semantic complexity since sentences that have patients (or non-core arguments) will tend to be more semantically complex than sentences which don't. This is consistent with the consensus in the literature that ERP components are markers for integration of meaning (Kemmerer, 2014). Another possible explanation for the similarity of ERP components to a patient-like profile is that the presence of modifiers might induce eye-movements (for example from the modifier to the modified argument), and the ERP components may partly be driven by or related to eye-movement planning. A third explanation is that patient-like words tend to occur later in sentences, and we might see these patient-like profiles as a proxy for sentence position (however, we note that sentence-length is not present in these profiles).

Eye-tracking and self-paced reading time tasks are similar to modifier and modifier-context tasks respectively. Figure 5.8 shows the task similarity profiles of the eye-tracking and self-paced reading time tasks. Eye-tracking tasks are highly similar to each other, particularly within corpus. They are all also anti-similar to PRP (personal pronoun) and the verb-context for semantic role ARG2 prediction

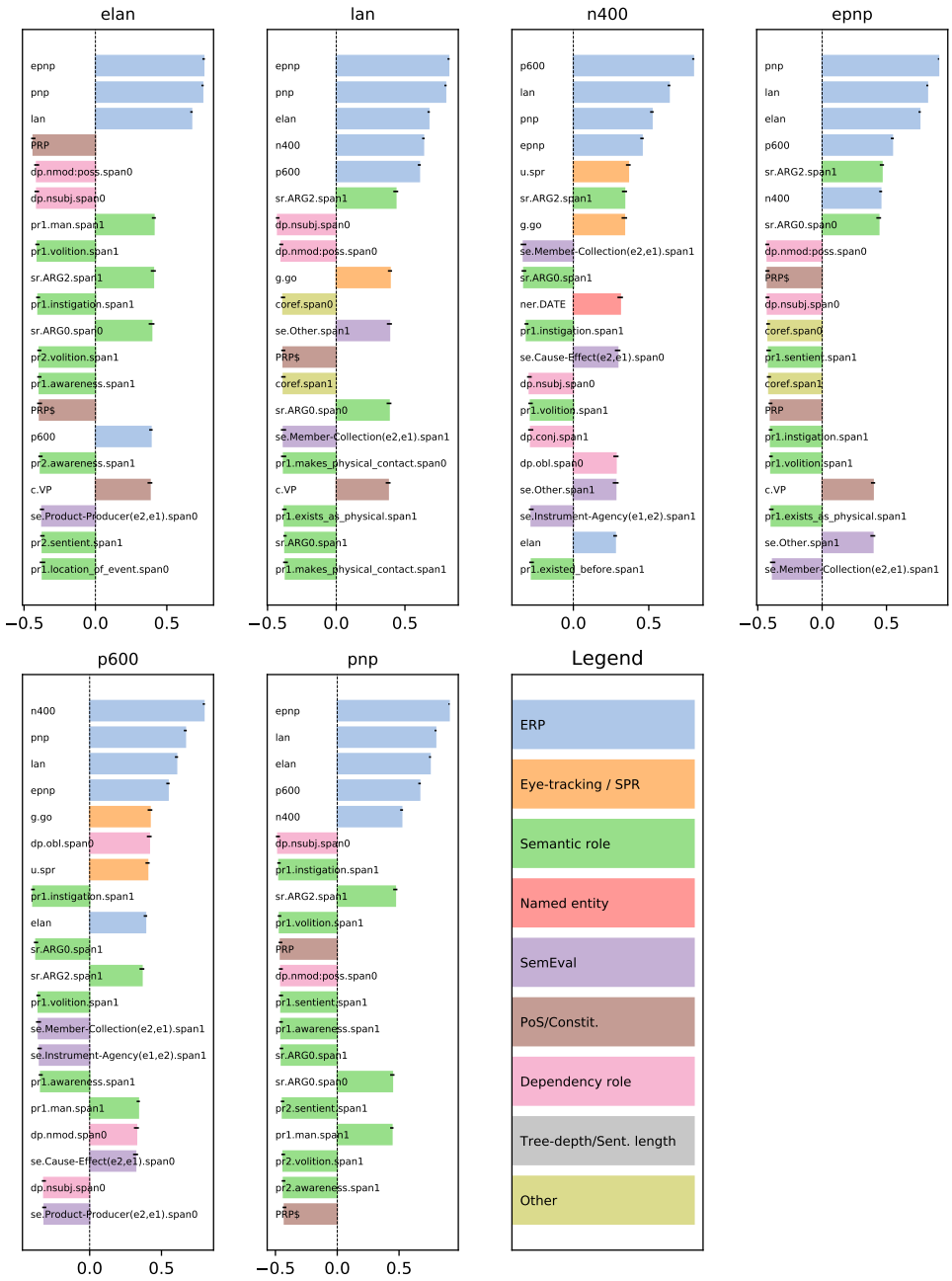


Figure 5.7: Task similarity profiles for each event-related potential (ERP) component prediction task. See text for discussion.

(sr.ARG2.span0, which is itself similar to PRP, dp.cop.span0, and VBZ) possibly because those words are all very short. Similarities also exist between eye-tracking tasks and compounds (a type of multi-word expression), NML (a type of multi-word expression), and conjucts, which one suspects may be because they induce eye-movements between their parts. Beyond this, we see similarities between eye-tracking tasks and adjective-like tasks — JJ (an adjective), ADJP (an adjective phrase), and VBN (a past participle, which can also be used in a modifier as in *he looked pleasant*) — and the sr.ARGM-MNR.span1 (the manner modifier to a predicate, usually an adverb), which may also induce eye-movements to the word or clause being modified. We also see similarity to VBG (a verb used as a gerund or present participle), which could be because verbs used as a present participle often introduce clauses, as in *I'm going to the store*. Self-paced reading time is similar to many of the eye-tracking tasks, and notably similar to ERP tasks. Although it doesn't have the same profile as the eye-tracking tasks, it follows a compatible pattern. It is similar to dp.cop.span1 (the governor in a copula, i.e. the complement, e.g. *Ivan is the best* ***dancer***), dp.compound.span1 (part of a multi-word expression), dp.nummod.span1 (a noun modified by a numerical expression), and pr1.location_of_event.span0 (a verb modified by a location argument). These all are the modifier-contexts (items being modified) in modifier relations or the subsequent words in multi-word expressions, and we hypothesize that participants dwell on these contexts as they integrate the meaning of the modifiers. Likewise, we suspect that dp.punct.span0 (a punctuation mark) induces a wrap-up effect during which the meaning of a phrase or sentence is being integrated. The anti-similarity of self-paced reading time to dp.obj.span1/sr.ARG1.span0 (the verb context in an object/patient relation) implies that when the model believes there is an increased probability of an object associated with the current verb, it also believes the self-paced reading time should decrease, which could suggest that people progress through transitive verbs to their objects quickly, but dwell longer on intransitive verbs as they process them.

fMRI voxel prediction is mostly orthogonal to our NLP tasks, but there is opportunity for further probing. Figure 5.9 shows the similarity profile for the summaries of participant F's regions of interest, where the summaries are computed as described in section 5.3.4 (additional participant profiles are available in appendix figures D.35-D.42). Despite the fMRI predictions being relatively good (see figures D.1 and D.2), the most notable aspect of the fMRI similarity profiles is the small scale of the similarities compared to other tasks. This is true not only for the summaries of fMRI similarities that we create, but also for individual voxels (see figure 5.11). If we instead take the absolute value of our task similarity measure before averaging over model initializations, we get a completely different result (figure 5.10). Before we address the absolute similarities, it is worth considering why this happens. This means that the low numbers we get from our original similarity computation are in part a result of the sign of the similarity between each individual voxel and the non-fMRI tasks changing from initialization to initialization. There could be multiple reasons for this. One plausible explanation is that the model is not really converging for individual voxels during training, and may oscillate around a local minimum. In this case, the sign of the similarity may change depending on where that oscillation is when training completes. This hypothesis is supported by the variance in correlations between predicted and actual values for a voxel in figure D.1. A second potential explanation is that the sign changes as a result of which parts of the data set are held out from run to run. There might be a different relationship between the fMRI data and the other tasks depending on how the data set is sampled. Understanding the source of the sign changes in similarity is an important point for follow up work, but the absolute task similarity profiles can nonetheless be informative. The absolute task similarity profiles (figure 5.10, additional participants' figures available in the appendix D.27-D.34) show that agent-like and patient-like proto-role property tasks are most similar to fMRI prediction and that sentence length tasks are also similar to fMRI prediction. We note that fMRI prediction does not seem to be driven by syntactic complexity (for example sentence length is present but not tree depth) or

Figure 5.8: Similarity profiles for eye-tracking tasks and self-paced reading time. See text for discussion

individual dependency role, part-of-speech, or constituency features (with the exception of the nominal subject). This suggests that the model is getting more of its leverage in predicting fMRI from semantics than from syntax, but we note that the absolute similarity numbers are still relatively low, and much of the information that the model is using to make predictions of fMRI is orthogonal to the information used by our set of non-fMRI tasks.

Opportunity in fMRI. While the scale of the similarities between fMRI tasks and other tasks is relatively low, the voxel-to-voxel similarities are much higher (see figure 5.11). The gap between the voxel-to-voxel similarities and the fMRI to non-fMRI similarities shows that most of the information the model is using to make predictions about fMRI is invisible to our NLP tasks. One potential explanation for this gap is that fMRI activity may reflect, to a large extent, the meaning of the individual words in a sentence as well as the meaning of the sentence itself. Our tasks on the other hand, primarily capture metadata about the sentence, such as the roles that the individual words play in a predicate. The proto-role properties may be the closest that our model comes to capturing the meaning of the words, and thus these may be represented most strongly in the similarity profiles. We note that with our method, we can refine the set of tasks that we include in our model to try to close this gap and to improve our hypotheses about what drives the fMRI recordings without making any further recordings. To improve our understanding, we need only to generate hypotheses and update the set of tasks in our model to further improve those hypotheses, then repeat this process until we capture the fMRI predictions with our set of tasks. In future work, we hope to close this gap.

Voxel-to-voxel similarities indicate more variation across participants than across regions. Turning to figure 5.12, we can use voxel-to-voxel similarities to investigate brain activity despite the low overlap with our NLP tasks. This figure shows a similarity matrix for the voxels that are relatively predictable (the correlation between predictions and actual values is at least 0.1), and that fall within a region of interest identified by the language network, according to the regions identified by (Fedorenko, Hsieh, et al., 2010). Note that these are not absolute similarities, but still have relatively large scale. The voxels are sorted first by participant (delineated by black grid lines), and then by region of interest within participant (delineated by gray grid lines). We note that voxels are more similar to each other within participant across regions than within region across participants. The figure also highlights three regions of interest by colors. Bright green shows the posterior cingulate, purple shows the dorsomedial prefrontal cortex, and black shows the angular gyrus. Portions of these these regions correspond to the blue stripes in the similarity matrix, so they seem to be differentiated from other regions by the model, and we also observe some similarity across participants within these regions. These regions possibly correspond to a subnetwork within the language network where processing is different from other regions.

5.5 Conclusion

In this chapter, we have proposed a method for analyzing task similarity in a multitask setting which does not require tasks to share examples, but instead learns a small number of latent functions over input text which can support all of the tasks, and relies on the degree to which tasks share the information in these functions to compute similarity. The method convincingly recovers congruent tasks from disparate datasets and finds high similarities between intuitively similar tasks. With the inclusion of proto-role properties, the similarity profiles produced by our method are also highly interpretable, and we have used these profiles to generate insights into the mechanisms used by the model to make predictions. For example, we hypothesize that the contextualized embedding of the verb is imbued with properties of the arguments to that verb, and

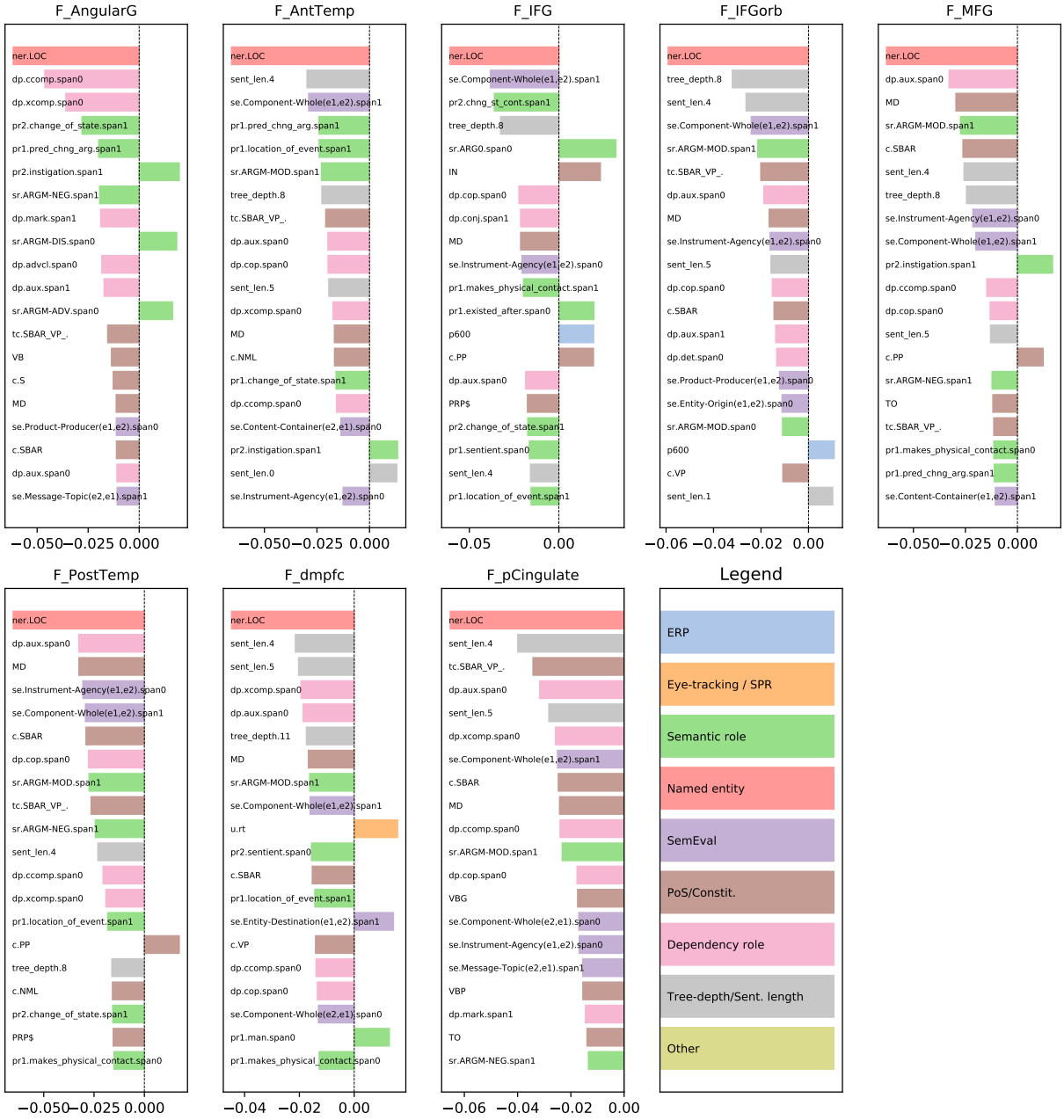


Figure 5.9: 20 largest magnitude task similarities for the summary of each region of interest for participant F as computed in section 5.3.4. Bar colors indicate the broad category of each task that fMRI is similar to.

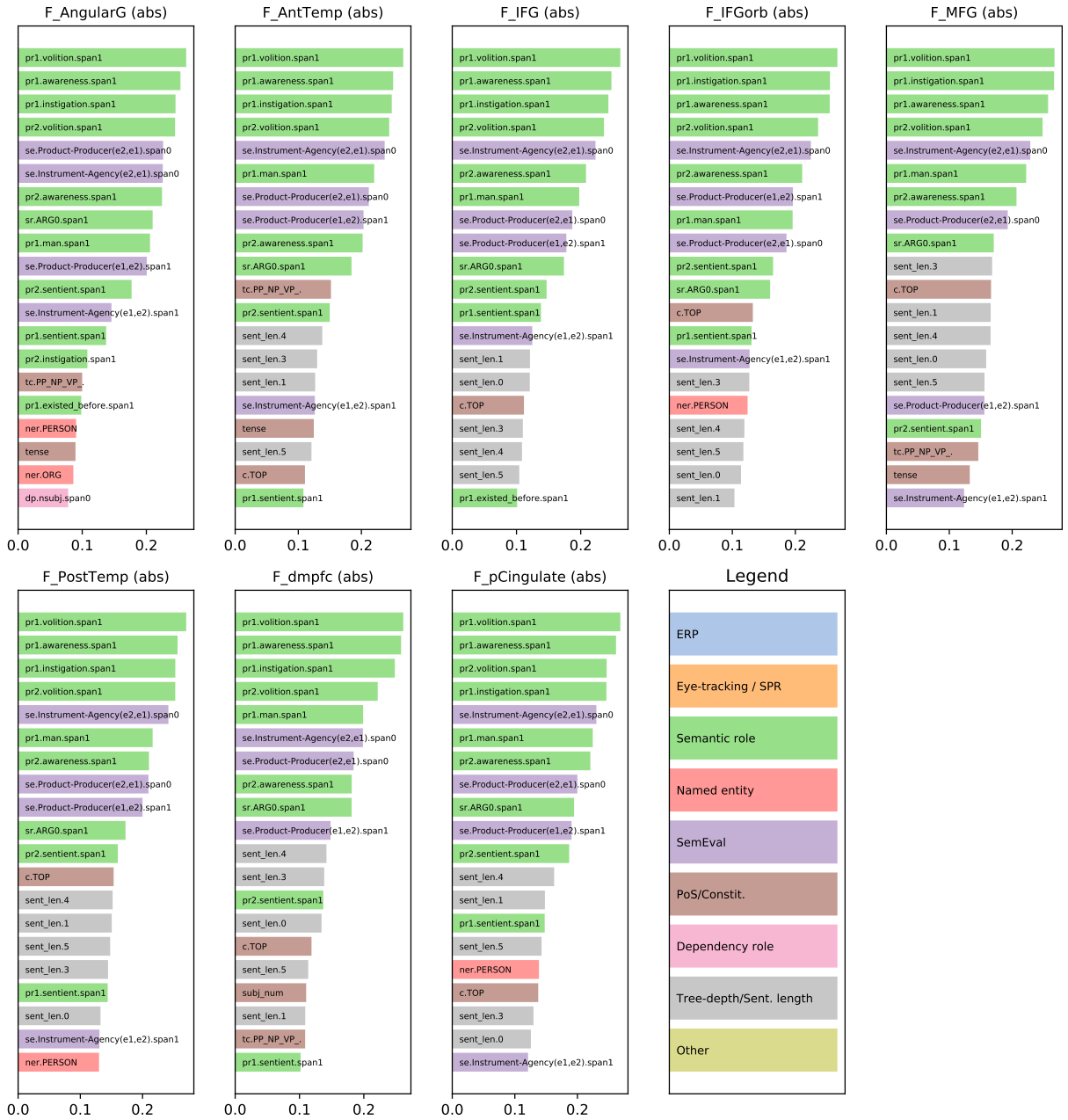


Figure 5.10: 20 largest absolute task similarities for the summary of each region of interest for participant F as computed in section 5.3.4. Bar colors indicate the broad category of each task that fMRI is similar to.

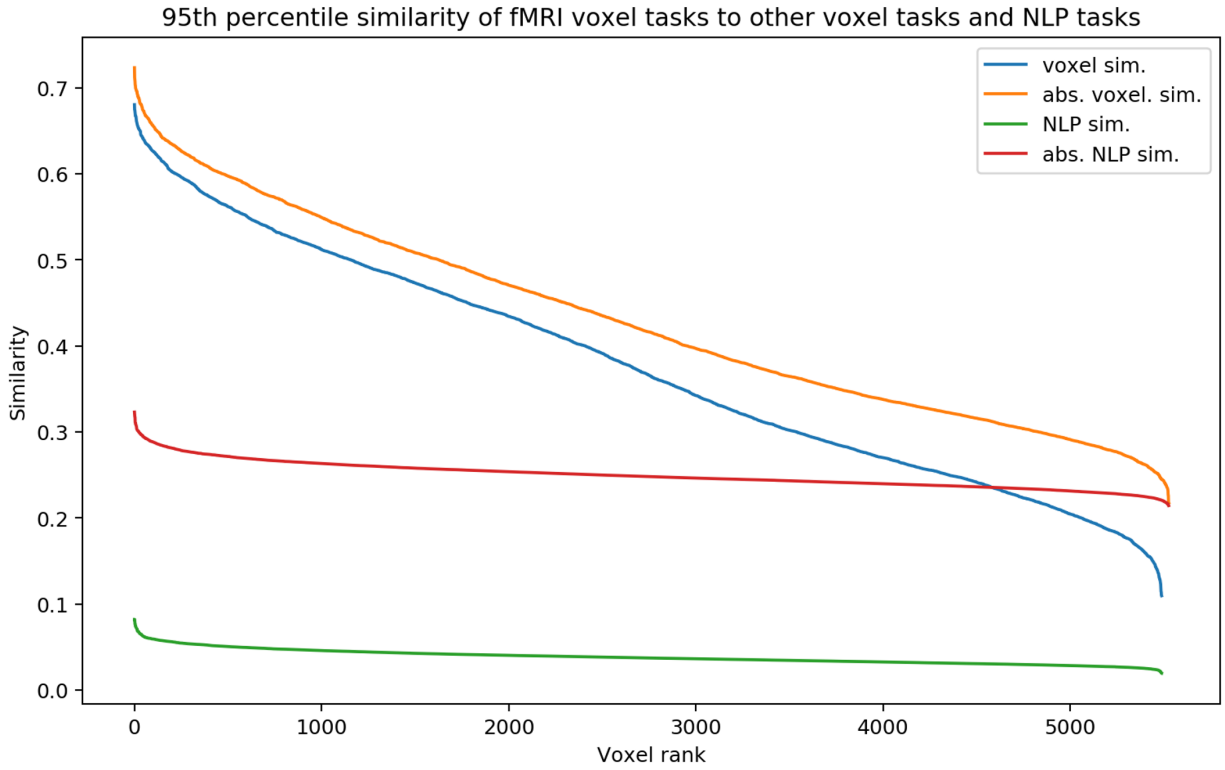


Figure 5.11: In this figure, for each voxel where the correlation between our model predictions and the actual values of a voxel are at least 0.1, we compute the $95th$ percentile similarity (and absolute similarity) of the weights associated with that voxel and the weights associated with all other voxels which can be predicted with a 0.1 correlation, along with the $95th$ percentile similarity (and absolute similarity) of the weights associated with that voxel and the weights associated with the 238 tasks and classes which are included in the similarity profiles. The similarities are then sorted (independently for each line in the plot) and plotted to show the gap between voxel-to-voxel similarities and NLP similarities.

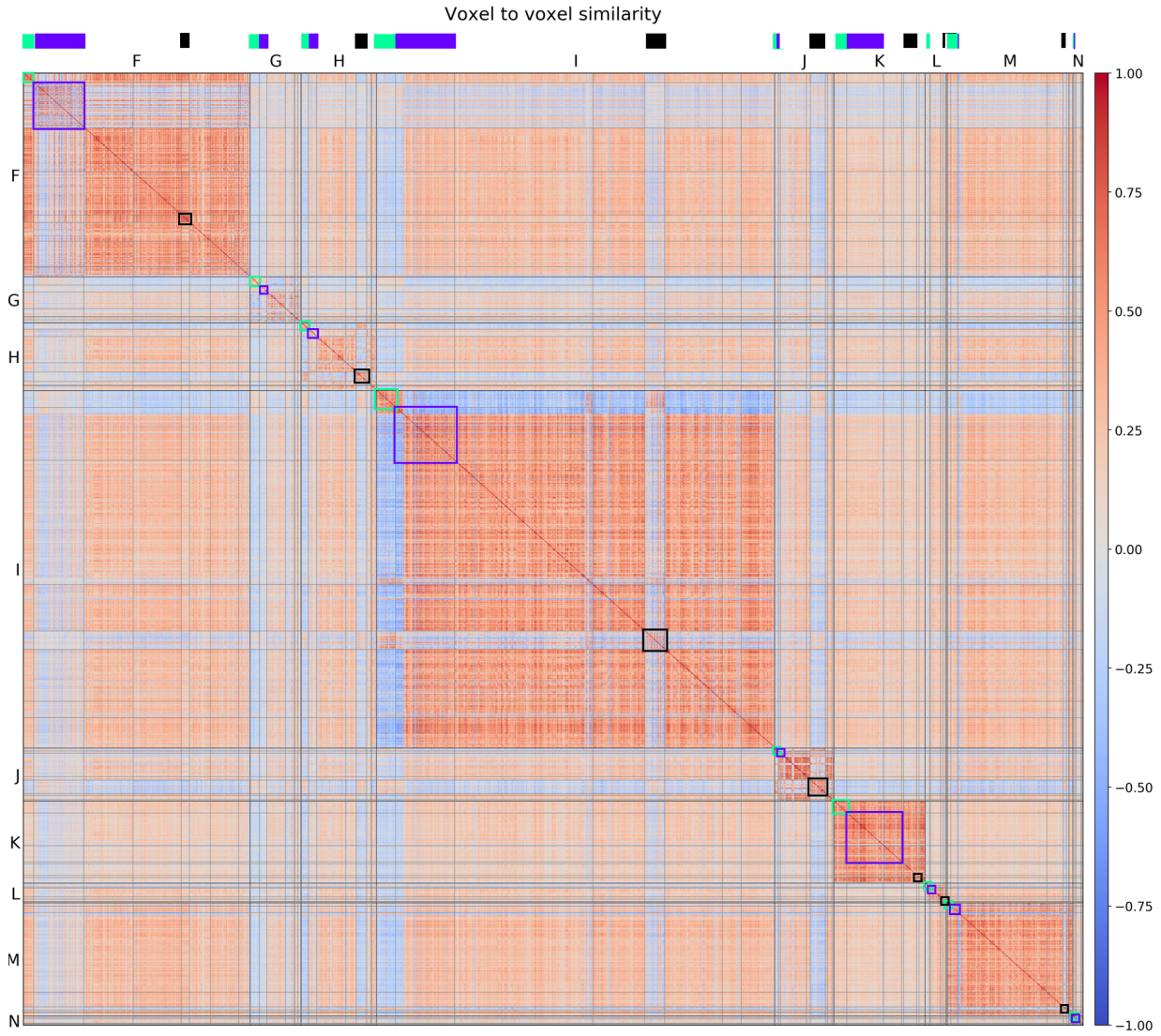


Figure 5.12: The similarity matrix for all voxels that meet two criteria. First the voxels are in one of the regions of interest identified with the language network using an updated version of the regions that are functionally defined in (Fedorenko, Hsieh, et al., 2010) and transformed to an individual participant's brain space using PyCortex (Gao et al., 2015), and second the correlation between the predicted and actual values for those voxels must be at least 0.1. Voxels are sorted first by experiment participant and then by region of interest within participant. Participants are delineated by black grid-lines in the similarity matrix, and regions are dilineated by gray grid-lines. Three regions are highlighted by colored bars at the top of the matrix showing the extent of the region and colored squares along the diagonal: the posterior cingulate is highlighted in bright green, the dorsomedial prefrontal cortex in purple, and the angular gyrus in black. This figure illustrates two points. First, voxels are much more similar to each other within participant across different regions than within region across different participants. Second, the voxels within the highlighted regions appear to be somewhat different to the voxels within other regions (note the blue stripes in the similarity matrix corresponding to portions of these regions), and there appears to be some similarity between voxels within these regions across participants.

that in some cases the model induces a competition between arguments to make its predictions by using that embedding. Our analysis also suggests that sequence-level embeddings contain detailed syntactic and semantic information about the tokens in the sequence, and that the model makes use of this low-level information for certain sequence-level tasks. Our main motivation for developing the method is to better understand how models make predictions for cognition-relevant tasks and ultimately what this tells us about cognition itself. To that end, we have applied our method to EEG, eye-tracking, self-paced reading, and fMRI data, resulting in somewhat interpretable similarity profiles for those tasks which can be viewed as an alternative to representational similarity analysis (RSA) (Kriegeskorte, Mur, and Bandettini, 2008). Our method can be applied to explore the cognitive processes involved in language comprehension by leveraging existing labeled datasets and deep language models without requiring new cognition-relevant data collection. This suggests a blueprint for future probing of the cognitive processes involved in language comprehension in which (1) cognition-relevant datasets are recorded while participants read or listen to naturalistic language, (2) a set of NLP tasks is selected to probe the cognition-relevant data, (3) the hyperparameters of the model are adjusted and the model is trained (in the future, this step could be automated through an AutoML framework), (4) hypotheses are generated from the similarity profiles our method generates, and steps 2-4 are repeated to continually refine and improve hypotheses about cognition without requiring new cognition-relevant datasets. In the current work, we find that the mechanisms our model uses to predict ERP components are most similar to proto-patient prediction, while the mechanisms for eye-tracking and self-paced reading time prediction are similar to predicting the existence of a modifier (for example an adjective or adverb) and the existence of a word or phrase being modified, along with prediction of punctuation (which we attribute to wrap up effects). These all seem to paint a picture that semantic complexity may be the main driver of the model's predictions of the cognition-relevant data. With respect to fMRI, our method is somewhat limited by noise in the data and/or modeling techniques. However, we find that the mechanisms of fMRI prediction can be at least partially characterized as being driven by proto-agent and proto-patient properties along with sentence length, consistent with the overall emerging picture. In future work, we hope to adjust the set of tasks we include in the model to better characterize fMRI prediction. We believe that this method is very powerful. New datasets and models can be added at any time to better characterize cognitive processes, and the method provides a partial solution for using black box models as a way to study cognition. We feel that this method is an exciting technique for computational cognitive neuroscience.

Chapter 6

Conclusion

6.1 Summary of Contributions

This dissertation develops methods for using multitask learning as an analysis tool, and applies those methods to better understand the cognitive processes underlying language processing in people. We recapitulate our contributions here. In chapter 3, we pioneer fine-tuning a deep language model to predict event-related potential (ERP) components. We show that fine-tuning can work in this setting, and that all of the ERP components are predictable. Furthermore, we bring together ERP components and behavioral data into a single model, allowing us to leverage heterogeneous datasets that are relevant to cognitive processing. We show that we can lower generalization error on predicting individual ERP task outputs by either training on multiple ERP tasks jointly, or by including behavioral data during training. As a first approach to using multitask learning as an analysis tool, we use constructive interference to explore relationships between ERP components. Using this method we find that the LAN and ELAN are related to the P600 (an expected result), that the LAN and ELAN are more broadly related to many ERP components, and that the N400 is relatively isolated from other components. We have suggested some hypotheses for these findings, including that the LAN and ELAN components might be primarily driven by working memory demands and that the N400 might be driven primarily by semantic complexity. Our findings in chapter 5 are somewhat different than our findings in 3 with respect to ERP components, and studying the cause of this difference is an opportunity for future research that we suggest below.

In chapter 4, we pioneer fine-tuning a deep language model to predict functional magnetic resonance imaging (fMRI) data. We show that a model which has been fine-tuned on fMRI data is better at predicting fMRI data than a model which has only been trained on a language modeling task (while this result might seem trivial, it need not be true — performance could be saturated before fine-tuning). We also show that if we fine-tune a model to predict one participant's brain activity, then that model is better able to predict other participants' brain activity than a model which has only been trained on a language modeling task. This is evidence that the way the model has learned to adjust its representations of language input captures information that is relevant to the prediction of many participants' brain activity, i.e. that it may be related to language processing in the brain. We explore the related question of whether a model which has been fine-tuned to predict magnetoencephalography (MEG) data is better at predicting fMRI data than a model which is fine-tuned on fMRI data alone, and we see mixed results. While it is better for some participants, it is also worse for other participants. We note, however, that for most participants language region prediction is improved even when overall prediction is not. Thus we have evidence that the changes in the parameters of the model when it is fine-tuned to predict brain activity generalize across participants, and we have weak evidence that these changes generalize across recording modalities, which indicates that those changes are

related to language processing and not the idiosyncracies of a particular participant or recording modality. This evidence supports our use of fine-tuned language models to analyze language processing in the brain, since it suggests that fine-tuning makes the representations in the model more similar to the representations in the brain. Finally, we suggest that we can explore the nature of the changes in the model parameters by looking at which language features are prevalent in examples where prediction accuracy changes the most after fine-tuning compared to the distribution of language features in the examples where prediction accuracy does not change. According to this method, we find that features or the model representations related to motion may be changing during fine-tuning, although the sample size is very small.

Lastly, in chapter 5, we develop a method for analyzing task similarity in a multitask learning setting based on model parameter similarity. By constraining a deep language model to produce a relatively small representation of the input which is shared by all tasks and by keeping the model as simple as possible between this shared layer and the task outputs, we can compute task similarity in terms of the parameters which map from this shared layer to the tasks of interest. We present a modified cosine similarity which we use on the model weights to measure task similarity in this setting, and we modify the MultiDDS algorithm of (Wang, Tsvetkov, and Neubig, 2020) to scale to our large number of tasks. We then bring together datasets from across disciplines (psycholinguistics, neuroscience, and natural language processing) to study cognitive processes using task similarity. Our analyses show that the method recovers as similar those tasks which are nearly the same by definition but come from different datasets, and tasks which are intuitively similar to each other across datasets. Using the similarity profiles of various tasks, we examine the mechanisms that the model is using to make its predictions. We hypothesize that the model constructs contextualized embeddings for verbs which contain information about the arguments in a predicate, and that in some cases the model uses competition between those arguments by comparing the information in the verb embedding to the information in a particular argument embedding. We also observe that the sequence-level embedding the model creates contains a lot of detailed information about the individual tokens in a sequence of text, and that the model makes use of this information for sequence-level tasks. Along with these general observations, we use the task similarities to better understand relationships between individual natural language processing tasks. The application of our method to analyses of ERP component tasks reveals that ERP prediction is similar to proto-patient role prediction, which suggests that part of the variance in ERP components is accounted for by the features of patient-like or non-core arguments in a proposition. This could be because sentences which contain patient-like or non-core arguments are more semantically complicated and increase cognitive load. We find that prediction of eye-tracking data is anti-similar to prediction of tasks that involve short words, and that eye-tracking prediction is also similar to prediction of modifiers in a sentence, such as adjectives. Self-paced reading time prediction is influenced by similar features, but where eye-tracking is more similar to the modifiers (adjectives and adverbs), self-paced reading time is more similar to the words or phrases being modified. We suggest that the dwell time on a word may increase when a participant is integrating the meaning of these modifiers, and that a similar wrap-up effect causes the self-paced reading time prediction to be related to punctuation prediction. Finally, we apply our method to fMRI data, and we find that prediction of fMRI is somewhat similar to prediction of agent-like and patient-like properties, along with prediction of sentence-length. However, the similarity scores are much lower than similarity of the prediction of one voxel to the prediction of other voxels. This gap suggests that we can continue to improve our understanding of what drives fMRI data by refining the set of tasks we include in our model, generating new hypotheses, and repeating this process of refinement without requiring any new brain activity recordings.

As a whole, this thesis examines how we can leverage the strengths of deep learning for the analysis of the cognitive processes involved in language processing in people. Deep learning allows us to combine together heterogeneous data to provide different kinds of evidence about language processing, and impor-

tantly, allows us to take advantage of large datasets for the analysis of cognition. The main weakness of deep learning is in its lack of interpretability, but we use multitask learning as an analysis tool to both gain insight into how deep learning models make their predictions and gain insight into what accounts for the variance in the observable data we have which is relevant to cognitive processes. We have developed and applied some methods to examine these processes, provided evidence that model parameters can capture information relevant to cognition, and, especially in chapter 5, we make significant progress towards a better understanding of language in the brain. We feel that the overall methodology is a very promising direction for computational cognitive neuroscience since it can naturally be applied to more and more datasets — which each provide additional evidence about cognitive processing or provide a new interpretable task to compare with that evidence — without a need to have overlapping examples or indeed any requirement apart from using text as input. We look forward to future work which expands on this promise.

6.2 Limitations

Throughout this thesis we have promoted multitask learning as a powerful technique for analyzing language processing in the brain. Our framework has several major advantages over other approaches. First, we can naturally combine datasets of starkly different sizes that do not share examples, requiring only that the examples are language related. Second, we can combine evidence from psycholinguistics, neuroscience, and natural language processing. Third, our results (especially from chapter 5) are interpretable. Finally, the method acts as a probe that can be easily extended by adjusting the set of tasks which are included. Despite these strengths, multitask learning is not without its limitations. We discuss a few of the limitations here, both for the design choices we use in the thesis work, and for the framework in general.

Forcing tasks to share information. A well-studied issue in the field of multitask learning is how to share information between tasks. If all of the tasks in a multitask problem are really just the same function between the input (a sequence of text in our case) and an output, then they can all be modeled by using a single scalar function of the input, and copying that scalar value as each task output prediction. In that case, multitask learning is equivalent to combining the examples from all of the tasks into a larger dataset, effectively increasing the number of samples. However, this is not the case for a set of interesting tasks. The tasks really are different functions, and they really need different intermediate representations. At the other extreme, we could allow the intermediate representations to be completely free (i.e. an infinitely wide representation), in which case the tasks no longer need to share any information. Our framework for using multitask learning as an analysis method is therefore asking to what extent a pair of tasks shares information given a certain trade-off between sharing of information between tasks and optimizing some multitask objective. This trade-off is intrinsic to multitask learning, so under any modeling choices, the task-similarities produced using the framework we have developed will always be conditioned on how that trade-off is made. Despite this, the task similarities that our model produces are optimized for a given trade-off, i.e. given a particular model architecture and language representation from pretraining, the method finds a locally optimal way to compress the information that supports tasks. Based on the low variance in task-similarities over model initializations, the manner in which the model compresses information robustly "collapses" the same pairs of tasks together. In our particular modeling choices, the amount of information the tasks must share is determined by the architecture of the model. The multitask objective is just the expected loss in chapter 3, and is an implicitly defined objective in chapter 5 which is a consequence of the MultiDDS algorithm variant we employ. Explicitly formulating a multitask objective to optimize for learning about language in the brain is left to future work that could make this trade off more principled.

Chapter	Contribution
3	We pioneer fine-tuning a deep language model to predict ERP components
3	We show multitask learning improves generalization error on ERP prediction
3	We combine behavioral data and neural activity prediction together in a single model, leveraging heterogeneous datasets
3	We use task-interference in multitask learning as a measure of task similarity to apply multitask learning as an analysis tool to ERP data
4	We pioneer fine-tuning a deep language model to predict fMRI data
4	We show that a fine-tuned model is better at predicting brain activity than a simple regression from a standard deep language model, i.e. that the model encodes information pertinent to brain activity into the parameters of the deep layers of the model
4	We show that the information encoded into a model's parameters when the model is fine-tuned to predict one participant's brain activity transfers to new participants, suggesting it is not idiosyncratic
4	We show that a model which has been fine-tuned on MEG data in some cases exhibits transfer to fMRI data, suggesting the parameters partially encode information about the relationship between language and brain activity, as opposed to the relationship between language and a recording modality
4	We use language feature distributions to interpret how a deep language model's representations change when it is fine-tuned to predict brain activity
5	We pioneer jointly training a deep language model on behavioral, neural, and natural language processing data to bring together heterogeneous datasets from multiple disciplines to inform our understanding of the cognitive processes involved in language processing in people
5	We use the parameters of a model to evaluate task similarity in a multitask learning framework
5	We develop the covariance-informed cosine similarity as a measure of similarity between model weights
5	We modify the MultiDDS sampling algorithm (Wang, Tsvetkov, and Neubig, 2020) to scale to a large number of tasks and to express preferences for particular tasks

Table 6.1: This table summarizes the methodological contributions in this thesis, and shows which chapters detail those contributions.

Chapter	Contribution
3	We find that the LAN and ELAN ERP components are both related to the P600 ERP component, and suggest this might have to do with syntactic complexity
3	We find more generally that the LAN and ELAN components are related to many other ERP components, and suggest that these relationships might be related to working-memory demands
3	We find that the N400 is relatively isolated from other ERP components and speculate that this might be because the N400 is related to semantic-retrieval while other ERP components may mark syntactic processes or later integration processes
3	We find that self-paced reading times benefit prediction of all ERP components, and eye-tracking data benefits prediction of many ERP components, perhaps signaling that all of these measures are to some extent driven by integration difficulty
4	We hypothesize that representations of sentences related to motion and imperative sentences may change the most when a model is fine-tuned to predict fMRI
5	We hypothesize that the model we develop in 5 embeds information about the arguments in a predicte into the representation of the verb, and that the decoders sometimes use that information to allow the arguments to "compete" for a particular label, such as ARG0 in semantic role labeling
5	We suggest that the model in 5 embeds information about many of the tokens in a sequence of text into the sequence-level representation for that sequence and that the model takes advantage of that detailed information for some sequence-level tasks
5	We find that there is overlap between semantic and syntactic tasks, with some semantic tasks, such as ARG1 relying primarily on syntactic clues for prediction, and some syntactic tasks, such as nsubj, relying heavily on semantic clues for prediction. However, for the most part syntactic tasks are similar to other syntactic tasks and semantic tasks are similar to other semantic tasks
5	We find that ERP components are all similar to each other. This result somewhat contradicts the analysis from chapter 3, and the difference could be from the model architecture or the similarity method
5	We find that ERP component prediction is similar to predicting patient-like properties in propositions
5	We find that eye-tracking prediction and self-paced reading time prediction are influence by word-length and by the presence of adjectives, adverbial phrases, and multi-word expresssions. These all may induce eye-movement, or potentially indicate increased meaning-integration demands. The latter hypothesis is somewhat supported by similarity between self-paced reading time and punctuation prediction, which we suggest may indicate a wrap-up effect
5	We find that fMRI prediction is quite different from most tasks we include in our set of NLP tasks, but is somewhat influenced by agent-like and patient-like properties of the words in a sentence, along with sentence-length

Table 6.2: This table summarizes the contributions this thesis makes to understanding language processing in both computers and humans, and shows which chapters detail those contributions.

Data starvation in cognition-relevant datasets. One of the main issues related to using machine learning to model relationships between language and neural or behavioral data is that for the complexity of the problem we are trying to model, the number of samples we have is far too small. Multitask learning can act as a regularizer by forcing tasks to share information, but this is only helpful if the tasks are really related. When they are not, forcing them to share information harms generalization error in the model. Small datasets still make the model susceptible to finding spurious relationships between language and cognition-relevant outputs, and we therefore still need to collect as much data as possible in cognition-relevant datasets. Our method enables these datasets to be based on naturalistic language stimuli that are not designed to test a specific hypothesis, and to some extent the regularization effect ameliorates the data starvation problem but it does not completely solve the data starvation problem.

Impact of architecture. The analysis in chapter 3 uses an LSTM architecture, while elsewhere in the thesis we use BERT. It is difficult to know the extent to which the choice of architecture impacts the results without running the experiments and optimizing the hyperparameter choices for each different architecture (a process we suggest automating in future work). A limitation of the conclusions we can make from the method we develop here is that some of the results could be a consequence of architecture choice. We speculate that for two models that have been sufficiently trained on large language corpora, and given that the two models use a relatively small bottleneck, the word-level task similarities would be somewhat insensitive to architecture choice. One thought experiment is to consider what would happen if we used all-but-one of the task predictions themselves as the basis for predicting the one other task. Then, if the models have similar performance on the all-but-one tasks, the weight vectors which combine those together should be similar for different choices of underlying architecture. A small bottleneck should be somewhat similar to using the task predictions as a basis, but with a bias. At the sequence level however, we suspect that the choice of architecture might have a larger impact. Models which use attention, like BERT, may have much different sequence embeddings than models which use recurrence, like an LSTM, and we suspect that this may impact the models' performance on sequence-level tasks, the similarity of sequence-level tasks to each other, and the similarity between fMRI prediction and other tasks.

Choice of task similarity measure. In chapter 3 we say that tasks are related if we see constructive interference when we train those tasks together, and in chapter 5 we use the covariance-informed cosine similarity to measure the similarity between model parameters. The particular design choice of the covariance-informed cosine similarity means that in chapter 5, we only capture linear relationships between tasks. This could be addressed by allowing non-linear transformations of the bottleneck layer or by changing the similarity metric to allow for non-linearity. More importantly though, both the constructive interference and covariance-informed cosine methods, and the philosophy of the thesis in general, consider the information overlap of tasks to be the definition of similarity. Indeed, the models we use throughout the thesis do not attempt to replicate the mechanisms of information transformation in the brain, but only to identify the information content. This approach misses certain relationships between tasks that we might otherwise identify. One could define two tasks as similar if those tasks use similar mechanisms. As an imperfect analogy, consider how three tasks might utilize resources in a computer — the first and second tasks might both use a lot of memory, make heavy use of a graphical processing unit (GPU), and never use the hard drive, while a third task might never use memory, use the central processing unit (CPU), and frequently use the hard drive. By some definition of similarity, we would ignore the information content, and call the first two tasks similar. Along these same lines, certain sensory experiences are similar to us as humans, e.g. the sour taste of a lemon is "sharp" like the edge of a knife or a high pitched note of music. The similarity between those sensory experiences has something to do with the acute temporal

nature of the perception and not (other aspects of) the information content. In principle these kinds of relationships could be identified under our definition of task similarity, but only with a model that is able to represent those aspects of a stimulus/input in some way. A second limitation of our approach is that because the approximations of the functions between the input and output are not explicitly defined, but rather implicitly defined by the parameters and architecture of the model, one cannot write down a simple expression to predict the output from the input. This makes definitive statements about the tasks difficult or impossible to make since the learned relationships are always a function of the datasets we use and are not defined apriori. However, confirmatory work could, in principle define the relationships explicitly.

6.3 Future Research Directions

Confirmation of hypotheses generated by our analyses. At several points in the thesis, we propose hypotheses about certain observations we make. For example, we propose that certain ERP components are related to working memory demands or that ERP components as a whole are related to patient-like properties in a sentence (note these hypotheses are not incompatible). In future work, we would like to expand on the analyses in this thesis, for example by considering the relative strength of the ERP components in comparison to the probability associated with specific semantic properties to see if ERP components can be predicted from those properties alone. We would also like to collect new data which can better discriminate between working memory demands and the presence of non-core arguments in sentences since these two phenomena will naturally be correlated in real world data.

Impact of architecture, pooling, and loss functions. As discussed in 6.2 the conclusions we can make from our model are somewhat limited by our inability to know which similarities are a consequence of model architecture. A related issue is that our particular choices of pooling functions and loss functions also impact task similarity outcomes. We would like to explore how different design choices for the architecture, pooling, and loss functions change the task similarities we compute to better tease apart these issues.

Understanding modeling mechanisms without a bottleneck. In chapter 5 we use a bottleneck layer to force different tasks to share information, and then we use task similarity to gain insight into mechanisms of model prediction. Of course, those mechanisms might be different if we had not imposed the bottleneck, and it would be nice to extend the task similarity method to a setting where we do not have to change the model in order to assess task similarity. This might be partially addressed by the covariance-informed cosine similarity since that metric compares weight vectors in a space where the components are orthogonal, but during training the model may learn to partition the components of its vectors so that they effectively are in separate spaces for each task (in an extreme version of this, given a large enough bottleneck size and enough training epochs, a model could memorize the dataset). Extension to this setting might be challenging.

Differentiating ERP components, regions-of-interest, and individual differences. The analysis methods we develop in chapters 3 and 5 seem to be somewhat "low resolution" in the sense that they find most ERP components are similar to each other and (in chapter 5) there is very little difference in the similarity profiles across regions and individuals. This might be intrinsic to reality, or it might be a consequence of the recording methodology, or it might be a consequence of the modeling. In the fMRI data, we see some evidence that we can differentiate regions of interest by removing the first few principle components of some regions from the activity of others — i.e., we see some evidence that after removing the background, a

given region is similar to the same region in other participants. It may therefore be possible to differentiate by improving the analysis methods or by collecting data from a larger number of participants.

Better learning on fMRI data. One of the main challenges in modeling fMRI data along with NLP tasks and even ERP data is the low signal-to-noise ratio in the fMRI data. Additionally, the large majority of voxels in the fMRI data are not predictable. We would like to let the model learn which voxels to predict and which not to predict. Treating each voxel as its own task rather than having a multidimensional fMRI task could be part of the solution, but then there are too many tasks to sample from as the model trains. Instead, this needs to be formulated as adjusting the loss / learning-rate / gradient per voxel. However, since fMRI overall has high variance compared to other tasks, this must be formulated to penalize bad voxels while encouraging the model to still learn fMRI as best it can. Additionally, we currently apply a single learning rate schedule for all tasks, but we would like to have a learning rate schedule per task, and possibly per voxel. Some techniques from the meta-learning community could probably be applied here to automatically adjust the learning rates, but this is an unusual setting that requires further research.

More tasks. From a neuroscience perspective, perhaps the most interesting line of work is to apply the methods developed here to additional tasks. We would like to include more neuroscience datasets, especially from naturalistic experimental paradigms, which cover diverse types of data (books, news, internet, etc.) both spoken and read. Additionally, the inclusion of additional natural language processing tasks has the potential to give us additional insight. One area in which the model in chapter 5 is lacking are tasks that capture word meaning. A language modeling task can capture word meaning and give us an upper bound on how much of the information the model uses to make its congition-relevant predictions can be captured through word-meaning tasks, but does not provide much other insight. Breaking down word meaning into subtasks would be more useful.

Appendices

Appendix A

Results from LSTM Training Variations

Target	Additional	POVE	Target	Additional	POVE	Target	Additional	POVE
ELAN		0.20	LAN		0.23	N400		0.20
ELAN	+ERP	**0.22**	LAN	+ ERP	**0.26**	N400	+ ERP	0.20
ELAN	+READ	**0.22**	LAN	+ READ	**0.25**	N400	+ READ	0.20
ELAN	+EYE	0.21	LAN	+ EYE	**0.24**	N400	+ EYE	0.18
EPNP		0.28	P600		0.24	PNP		0.28
EPNP	+ ERP	0.28	P600	+ ERP	**0.25**	PNP	+ ERP	**0.31**
EPNP	+ READ	**0.29**	P600	+ READ	**0.25**	PNP	+ READ	**0.30**
EPNP	+ EYE	0.29	P600	+ EYE	0.24	PNP	+ EYE	0.29

Table A.1: Proportion of variance explained for each of the ERP components when using only the forward direction of the encoder (mean of 100 training runs). +ERP indicates the best combination of ERP training signals for the target ERP component, + READ indicates the inclusion of self-paced reading times, +EYE indicates the inclusion of eye-tracking data, and bold font indicates a significant difference from training on the target component alone.

Here we present a visualization (Figure A.1) of the results presented in Table 3.1 of chapter 3, and a visualization (Figure A.2) of a more complete set of results from which the information in table 3.2 of chapter 3 is drawn. We also show supplemental results for variants of our primary analysis on multitask learning with eye-tracking, self-paced reading time and ERP data. In the variants we modify the input representation to our decoder network to see whether the relationships between the behavioral data and neural activity appear to be consistent with different choices of encoder architectures. Additional (and more varied) choices or architectures are left to future work. The results in Table A.1 reflect using only the forward-encoder (rather than the bi-LSTM) in the encoder network, while the results in Table A.2 reflect using only the word embeddings (i.e. bypassing the LSTM entirely). While the results are clearly worse for each of these choices of architecture than for using a bi-LSTM encoder, the relationships between the behavioral data and the ERP signals is qualitatively similar. Finally, A.3 shows the Pearson correlation coefficient between different measures. We note that the patterns of correlation are different than the patterns of which measures benefit from joint training with each other.

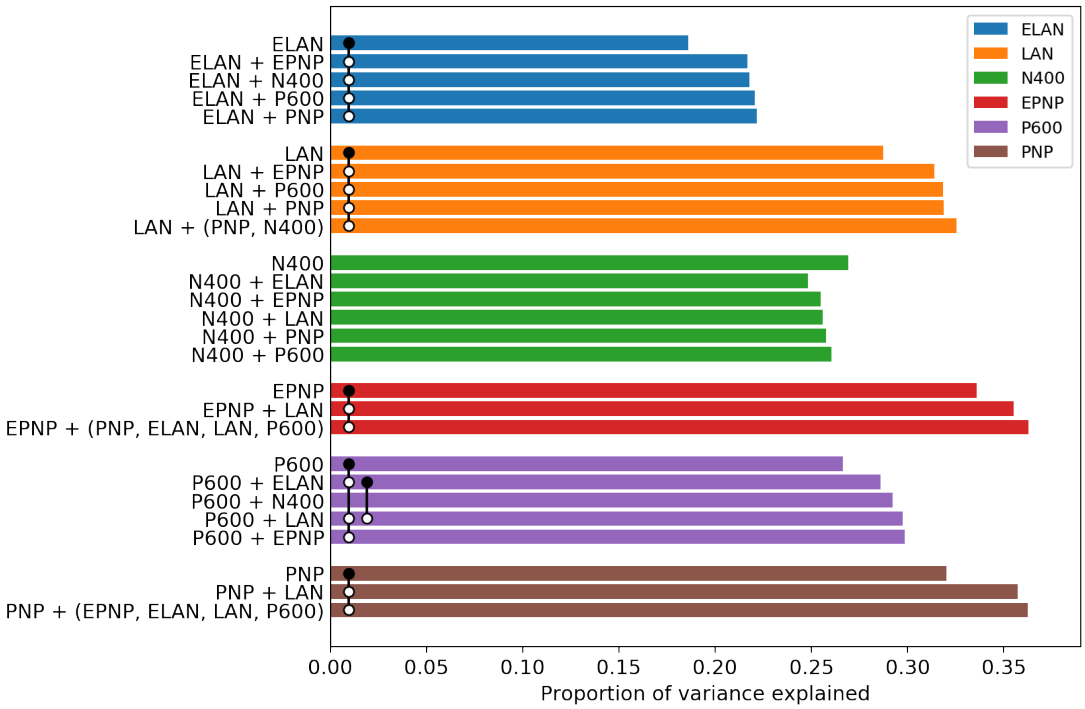


Figure A.1: The proportion of variance explained for prediction of each of the ERP signals (mean of 100 training runs). The target ERP is indicated by color; each group of bars shows performance for a different target ERP. The top bar in each group shows the proportion of variance explained when the model is trained using only the target ERP. The bottom bar in each group shows the maximum proportion of variance explained over all combinations of training ERPs (or in the case of the N400, the second best). Also shown in each group are any training combinations that (i) used no more than the number of ERP signals used by the combination that achieved the maximum, and (ii) which were not significantly different from the maximum. Bars are statistically different from each other if a black dot on one bar is connected by a contiguous vertical line to a white dot on the other bar. The bars in the N400 group are not significantly different from each other. The N400 signal is best predicted when the model is trained on just that signal. In every other group, there is at least one ERP that, when combined with the target ERP during training, improves the prediction of the target ERP. The results suggest that these pairs are related: (LAN, P600), (LAN, EPNP), (LAN, PNP), (ELAN, N400), (ELAN, EPNP), (ELAN, PNP), (ELAN, P600), (EPNP, P600).

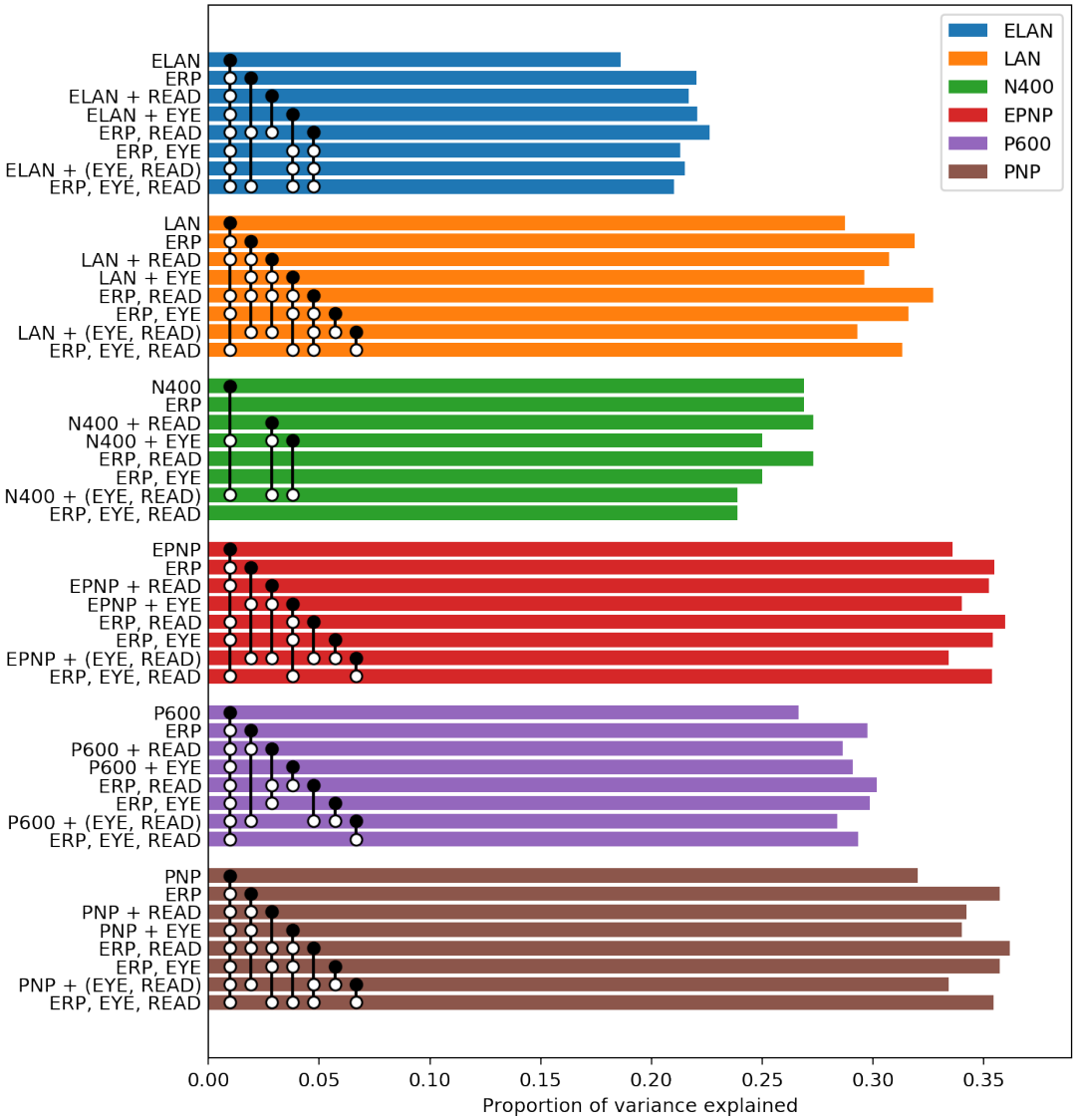


Figure A.2: The proportion of variance explained for prediction of each of the ERP signals (mean of 100 training runs). The target ERP is indicated by color; each group of bars shows performance for a different target ERP. The top bar in each group shows the proportion of variance explained when the model is trained using only the target ERP. Moving down, the next bar in each group, labeled *ERP* shows the proportion of variance explained by the best combination of ERP signals for the target ERP. The other bars in each group moving from top to bottom show training variations that use behavioral data with either just the target ERP, or with the best combination of ERP signals. *READ* denotes self-paced reading data, and *EYE* denotes all four eye-tracking measures (in this analysis we use right-bounded pass time, gaze duration, go-past time, and first-fixation duration). Pairs of bars are significantly different from each other (paired t-test, false discovery rate ¡ 0.01) if a black dot on one bar is connected to a white dot on the other bar by a contiguous vertical line. Self-paced reading time benefits prediction of all target ERP components except the N400. In the case of the ELAN, LAN, and PNP, self-paced reading time also has marginal benefit compared to the best combination of ERP training signals. Eye-tracking data benefits prediction of the ELAN, P600, and PNP components.

Target	Additional	POVE	Target	Additional	POVE	Target	Additional	POVE
ELAN		0.15	LAN		0.17	N400		0.05
ELAN	+ ERP	**0.18**	LAN	+ ERP	**0.19**	N400	+ ERP	0.05
ELAN	+ READ	**0.18**	LAN	+ READ	**0.19**	N400	+ READ	0.08
ELAN	+ EYE	**0.19**	LAN	+ EYE	**0.19**	N400	+ EYE	0.10
EPNP		0.18	P600		0.10	PNP		0.20
EPNP	+ ERP	**0.20**	P600	+ ERP	**0.13**	PNP	+ ERP	**0.23**
EPNP	+ READ	**0.20**	P600	+ READ	**0.13**	PNP	+ READ	**0.22**
EPNP	+ EYE	**0.21**	P600	+ EYE	**0.14**	PNP	+ EYE	**0.23**

Table A.2: Proportion of variance explained for each of the ERP components when using only the word embeddings as input to the decoder and bypassing the LSTM entirely (mean of 100 training runs). +ERP indicates the best combination of ERP training signals for the target ERP component, + READ indicates the inclusion of self-paced reading times, +EYE indicates the inclusion of eye-tracking data, and bold font indicates a significant difference from training on the target component alone.

Signal	ELAN	EPNP	LAN	N400	P600	PNP	FIX	PASS	GO	RIGHT	READ
ELAN	1.00	0.27	0.32	0.11	0.10	0.24	0.27	0.26	0.27	0.26	-0.04
EPNP	0.27	1.00	0.66	0.41	0.50	0.83	0.17	0.17	0.19	0.17	0.02
LAN	0.32	0.66	1.00	0.58	0.33	0.47	0.12	0.11	0.13	0.12	0.01
N400	0.11	0.41	0.58	1.00	0.47	0.33	-0.04	-0.04	-0.02	-0.03	0.11
P600	0.10	0.50	0.33	0.47	1.00	0.69	0.14	0.14	0.16	0.14	0.10
PNP	0.24	0.83	0.47	0.33	0.69	1.00	0.25	0.24	0.26	0.25	0.03
FIX	0.27	0.17	0.12	-0.04	0.14	0.25	1.00	1.00	1.00	1.00	0.04
PASS	0.26	0.17	0.11	-0.04	0.14	0.24	1.00	1.00	1.00	1.00	0.04
GO	0.27	0.19	0.13	-0.02	0.16	0.26	1.00	1.00	1.00	1.00	0.05
RIGHT	0.26	0.17	0.12	-0.03	0.14	0.25	1.00	1.00	1.00	1.00	0.04
READ	-0.04	0.02	0.01	0.11	0.10	0.03	0.04	0.04	0.05	0.04	1.00

Table A.3: Raw Pearson's correlation coefficients (computed on content words after the standardization and participant-averaging) between each neural and behavioral measure and each other measure. FIX indicates first-fixation time, PASS indicates first-pass time, GO indicates go-past time, RIGHT indicates right-bounded reading time, and READ indicates self-paced reading. Many of the measures are highly correlated, but the pattern of correlations is different from the pattern of benefits that we find during joint-training. In particular we note that the N400 is correlated with the other ERP signals, and yet we do not see benefit in prediction of the N400 when jointly training a model to predict it and other signals.

Appendix B

Additional Views of Model Comparisons in Generalizability of Models

B.1 Additional views of voxel-level comparisons

In chapter 4, Figure 4.3 shows a summary of spatial distributions of changes in fMRI prediction accuracy after fine-tuning by showing lateral views of the left hemisphere of all nine experiment participants across three different models. In Figures B.1, B.2, and B.3 we break out the three different models into separate figures and include the right hemisphere and medial views for each participant.

B.2 Model comparison using proportion of variance explained

Although we believe that the 20 vs. 20 accuracy described in section 4.3.4 (Mitchell et al., 2008; Wehbe, Vaswani, et al., 2014; Wehbe, Murphy, et al., 2014) gives a more intuitive comparison of models than the proportion of variance explained, both metrics have value. In some ways the proportion of variance explained is more sensitive to changes since the accuracy quantizes the results. Figure B.4 shows the same results as Figure 4.2 from section 4.4, but in terms of proportion of variance explained rather than 20 vs. 20 accuracy. The results are qualitatively similar, but we can even more clearly see the effects of overfitting in the models as the proportion of variance explained becomes negative when we include all voxels in the mean difference.

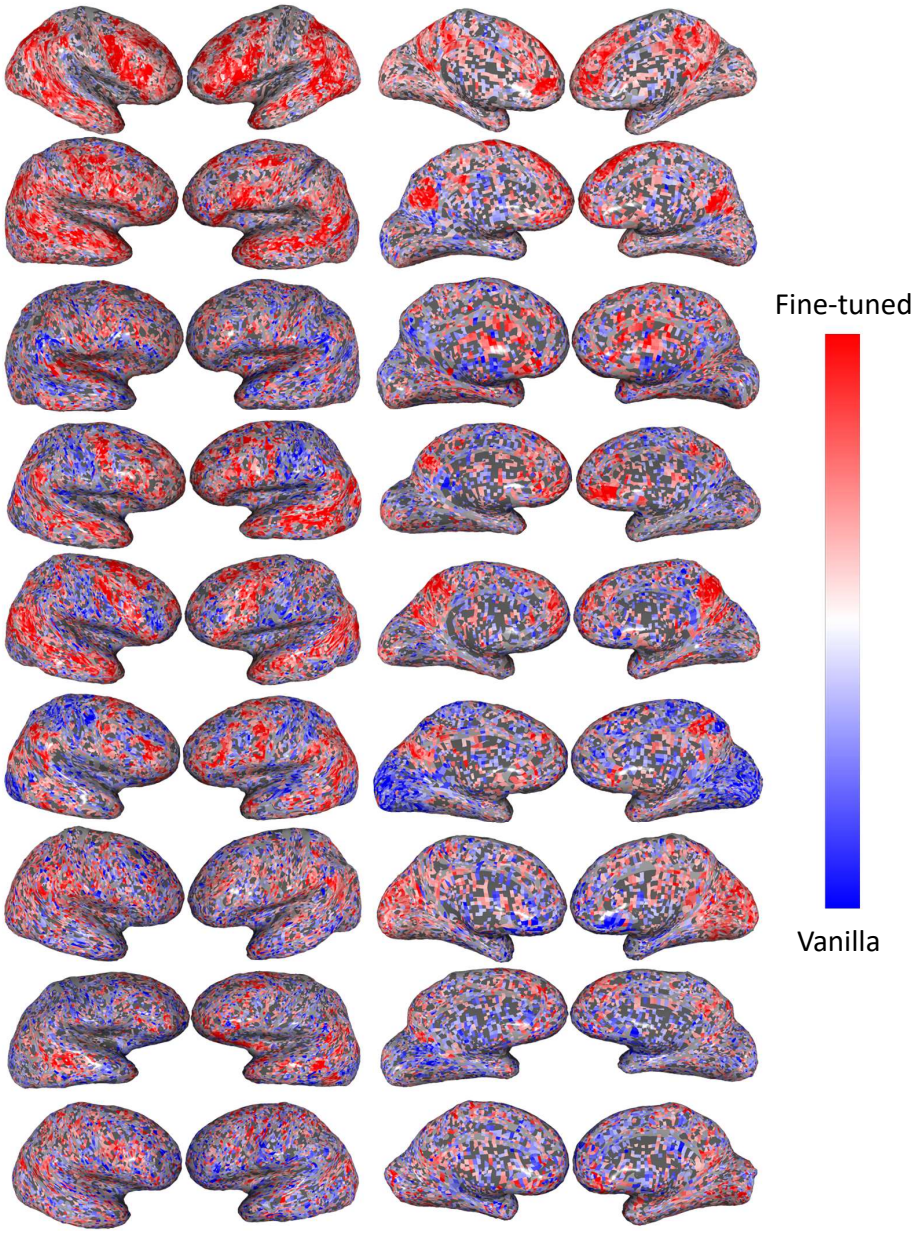


Figure B.1: Comparison of accuracies on the 20 vs. 20 classification task between the fine-tuned and vanilla models (described in section 4.3.4) at a voxel level for all 9 participants we analyzed. Moving from left to right across the page, the columns show inflated lateral views of the right and left hemisphere followed by inflated medial views of the left and right hemisphere respectively. Each row shows one participant. Only voxels which were significantly different between the two models according to a related-sample t-test and corrected for false discovery rate at a .01 level using the Benjamini–Hochberg procedure (Benjamini and Hochberg, 1995) are shown. The color-map is set independently for each of the participants such that the reddest value is at the $95th$ percentile of the absolute value of the significant differences and the bluest value is at the negative of this reddest value. The fine-tuned model tends to have higher prediction accuracy than the vanilla model in language areas.

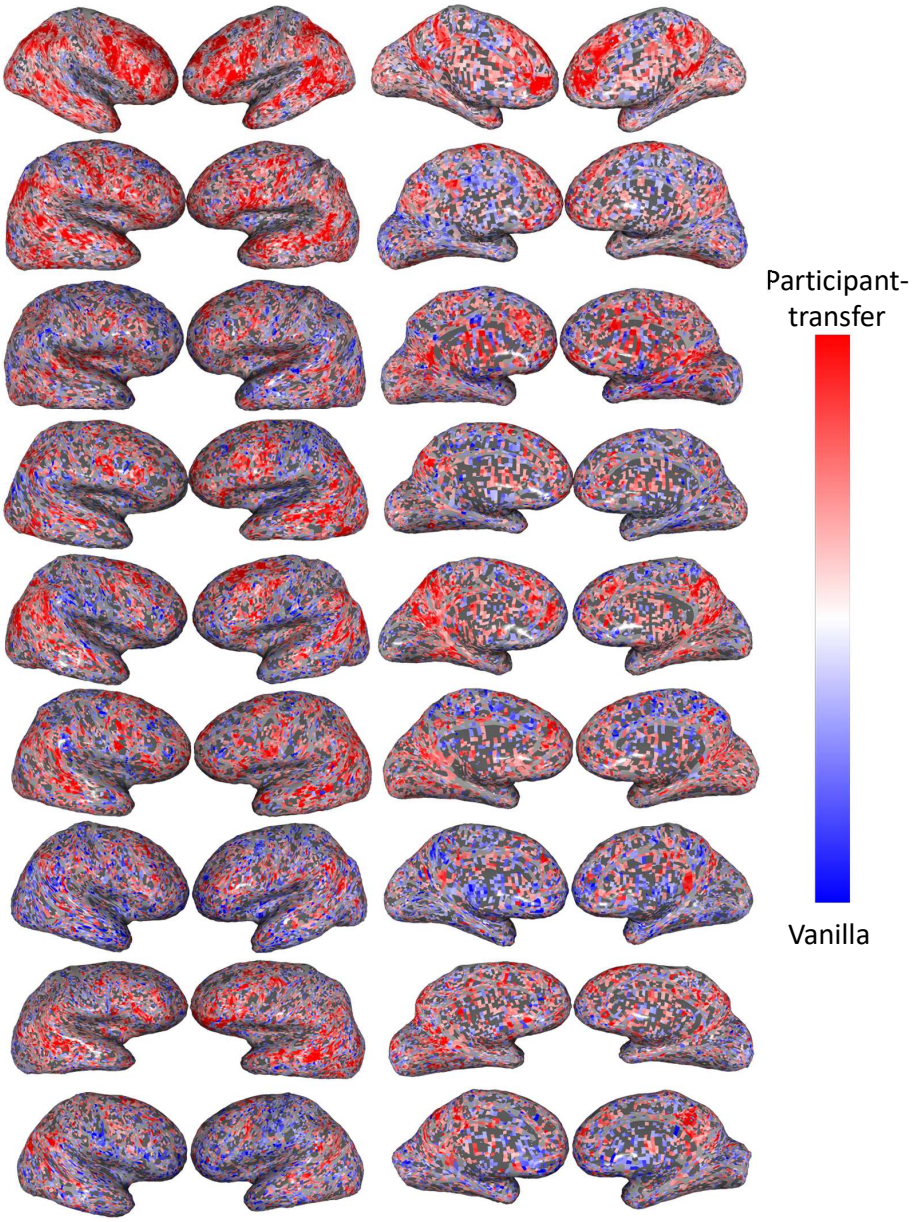


Figure B.2: Comparison of accuracies on the 20 vs. 20 classification task between the participant-transfer and vanilla models (described in section 4.3.4) at a voxel level for all 9 participants we analyzed. Moving from left to right across the page, the columns show inflated lateral views of the right and left hemisphere followed by inflated medial views of the left and right hemisphere respectively. Each row shows one participant. Only voxels which were significantly different between the two models according to a related-sample t-test and corrected for false discovery rate at a .01 level using the Benjamini–Hochberg procedure (Benjamini and Hochberg, 1995) are shown. The color-map is set independently for each of the participants such that the reddest value is at the $95th$ percentile of the absolute value of the significant differences and the bluest value is at the negative of this reddest value. Like the fine-tuned model, the participant-transfer model tends to have higher prediction accuracy than the vanilla model in language areas.

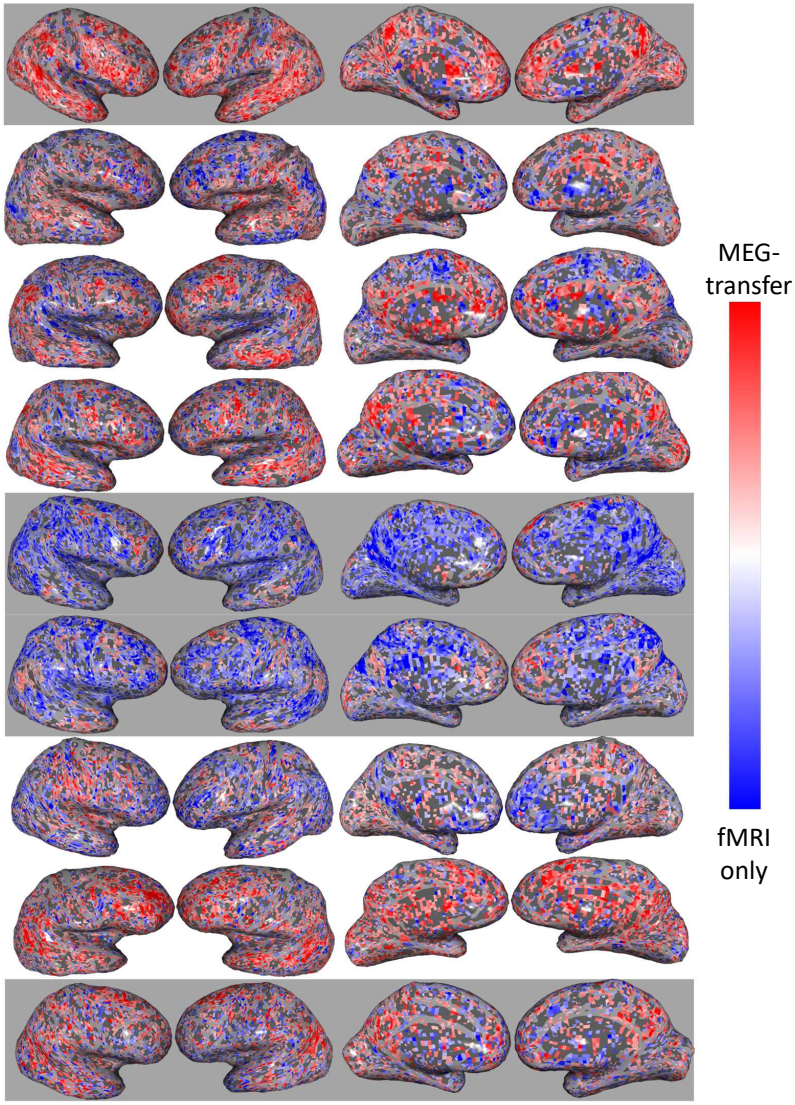


Figure B.3: Comparison of accuracies on the 20 vs. 20 classification task between the MEG-transfer and fMRI only models (fMRI only is referred to elsewhere as the fine-tuned model, both models are described in section 4.3.4) at a voxel level for all 9 participants we analyzed. Moving from left to right across the page, the columns show inflated lateral views of the right and left hemisphere followed by inflated medial views of the left and right hemisphere respectively. Each row shows one participant. Rows with a grey background indicate participants whose data were used in both the MEG training and the fMRI training (i.e., these participants were scanned in separate sessions in both modalities). Only voxels which were significantly different between the two models according to a related-sample t-test and corrected for false discovery rate at a .01 level using the Benjamini–Hochberg procedure (Benjamini and Hochberg, 1995) are shown. The color-map is set independently for each of the participants such that the reddest value is at the $95th$ percentile of the absolute value of the significant differences and the bluest value is at the negative of this reddest value. For most participants, prediction of language areas improves with the MEG-transfer model. However, the results are mixed and for some participants, prediction in (some) language areas is arguably worse. Nonetheless, the results suggest that the changes to the model learned during training to predict MEG are helpful for predicting fMRI data in language areas.

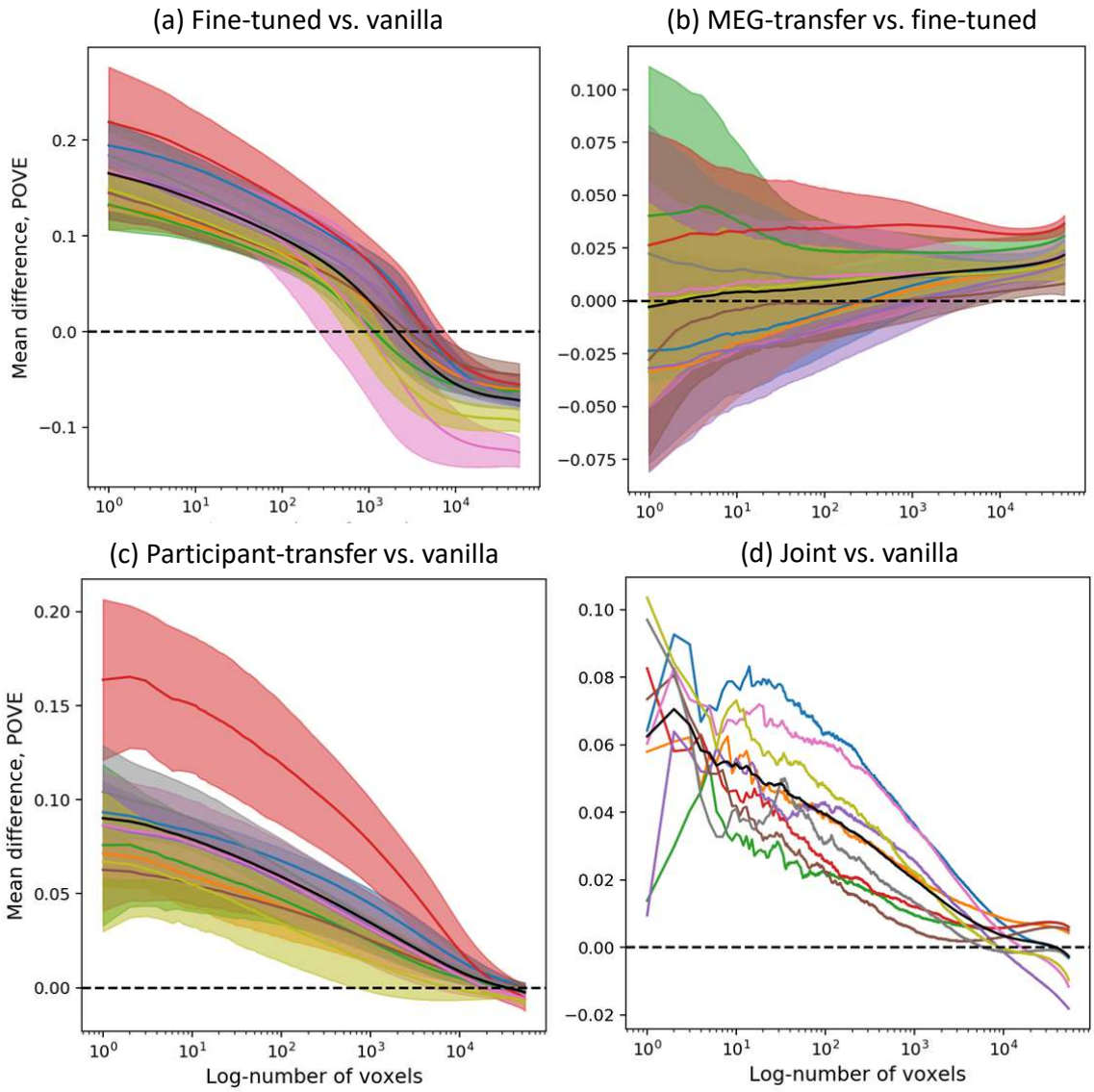


Figure B.4: Comparison of accuracies of various models. In each quadrant of the figure above, we compare two models. Voxels are sorted on the x-axis in descending order of the maximum of the two models' proportion of variance explained. The colored lines (one per participant) show differences between the two models' mean proportion of variance explained, where the mean is taken over all voxels to the left of each x-coordinate. In (a)-(c) Shaded regions show the standard deviation over 100 model initializations – that computation was not tractable in our framework for (d). The black line is the mean over all participants. In (a), (c), and (d), it is clear that the fine-tuned models are more accurate in the prediction of voxels than the vanilla model for a large number of the voxels. In (b), the results are more mixed. The MEG-transfer model seems to have roughly the same proportion of variance explained as a model fine-tuned only on fMRI data, but in Figure B.3 we see that in language regions, for most participants, the MEG-transfer model appears to be more accurate.

Appendix C

Additional Modeling Details for the Model Parameter Similarity Method

This appendix provides additional details about the methods in chapter 5 which are not considered critical to understanding the chapter, but which are important to reproducing the results or which may enrich the understanding of the interested reader.

C.1 Task preprocessing and train-test splits

For the cognition-relevant tasks in chapter 5, we apply some preprocessing steps before training our model. We also have somewhat more complicated schemes for splitting the data into a training set, a task-sample-rate tuning set, and a validation set than for the natural language processing (NLP) tasks. For NLP tasks, the splits are predefined and we reserve 20% of the training data for the task-sample-rate tuning set. The examples selected for this 20% are constant across model initializations.

fMRI tasks. functional magnetic resonance imaging (fMRI) data are recorded while participants read a chapter of *Harry Potter and the Sorcerer's Stone* Rowling, 1999, which is available online[1] (Wehbe, Murphy, et al., 2014). The story is shown to a participant word-by-word in an fMRI scanner, with the participant seeing one word every 0.5 seconds. One fMRI image is captured every 2 seconds. The fMRI data are recorded from 9 participants in 4 sessions. Each time we initialize our model, we take the modulus of the index of the initialization with 4 to create a cross-validation id. This id is used to select which one of the 4 sessions is held out during training. A second session is reserved as a task-sample rate tuning set to adjust how often batches from a specific task are chosen during training (see section 5.3.4). The other two sessions comprise the primary training data for that model initialization. Each session is independently detrended and standardized such that the mean over samples is 0 and the standard deviation is 1. We use mean squared error (MSE) as the loss for fMRI data.

ERP tasks. Event-related potential (ERP) tasks are derived from electroencephalography (EEG) data recorded on sentences from the University College London (UCL) corpus (Frank, Otten, et al., 2015). There are six tasks in this data for each of the six ERP components considered by Frank, Otten, et al. (2015) — the N400, P600, EPNP, PNP, ELAN, and LAN. These ERP components have been associated with language specific processes modulated by semantic and syntactic complexity, as well as with non-language specific factors such as working-memory demands and demands on attentive processing. The original paper associated with the dataset has additional details about the ERP components (Frank, Otten, et al., 2015), and several good reviews of these ERP components are available (Kemmerer, 2014; Kutas and Federmeier, 2011; Kuperberg et al., 2003; Van Petten and Luka, 2012), see also chapter 2. In this dataset, each ERP component is computed by taking the average of a predefined set of sensors within a predefined window of time, where the set of sensors and the time-window are specific to each ERP component (see figure C.1). This gives one scalar value per word per participant for each of the six ERP tasks. A baseline value is also provided per participant per word per ERP component. To preprocess the data, we first subtract the baseline values from the ERP component values, then we compute the mean and standard deviation of the values on the content words in the training data for each participant. We subtract out this mean and rescale by the standard deviation such that the mean over the content words for a participant becomes 0 and the standard deviation becomes 1. Next, we average over the participants, leaving us with one scalar value per ERP component per word. We then compute the mean and standard deviation over these averaged values in the training data for the content words, and remove them so that the final values have mean 0 and standard deviation 1. We apply a similar process to the held out sets, but we use the means and standard

[1] http://www.cs.cmu.edu/~fmri/plosone/

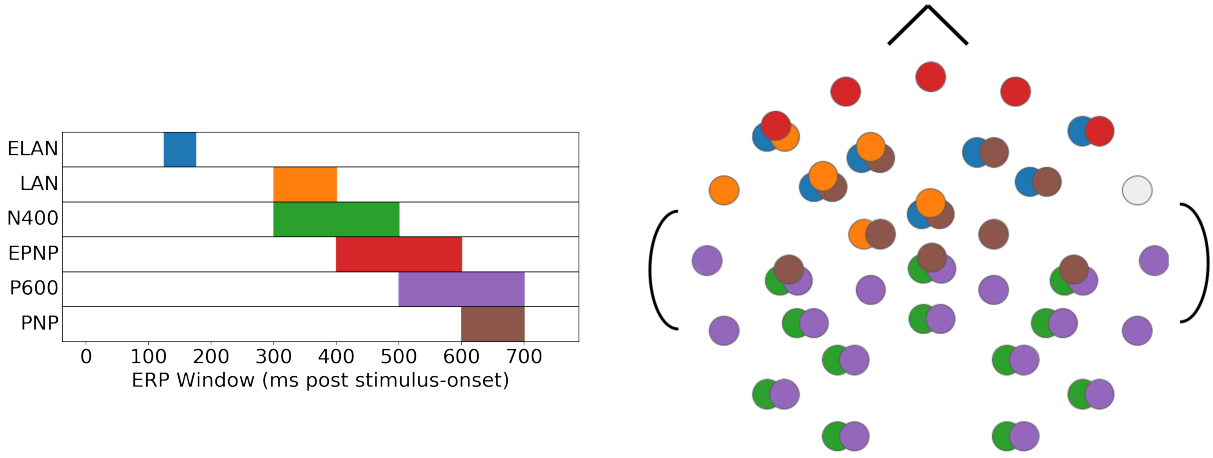


Figure C.1: The electrodes from which each event-related potential was recorded in the data from Frank, Otten, et al. (2015) (after figure 3 in (Frank, Otten, et al., 2015)). The bottom portion of the figure shows a top-down schematic of the electrode locations with the nose facing towards the top of the page. Each ERP is the mean potential from all of the indicated electrodes during a specific time-window, creating a single scalar value per ERP per word. Overlapping circles indicate multiple ERPs recorded from the same electrode. The ELAN is measured from 125-175ms after stimulus onset, the LAN from 300-400ms, the N400 from 300ms-500ms, the EPNP from 400-600ms, the P600 from 500-700ms, and the PNP from 600-700ms.

deviations computed on the training data when we center and rescale. MSE is used as the loss for ERP data. In our model, we make a prediction for each token in a sequence of text, but we only train and evaluate on the first token of each content word in the sequence (we consider any word that is an adjective, adverb, auxiliary verb, noun, pronoun, proper noun, or verb (including to-be verbs) to be a content word). For each initialization of our model, we randomly select 10% of the sentences in the data to hold out as a validation set and 10% of the remaining sentences for the task-sample-rate tuning set (see section 5.3.4). The remainder of the data becomes the primary training data. Note that along with ERP data, we also use eye-tracking data and self-paced reading time on the UCL corpus. The train-test split is done at the corpus level, such that if a sentence is in the validation set, it is in the validation set for all types of data associated with that sentence.

Eye-tracking tasks. In addition to the ERP data on the UCL corpus, we also include eye-tracking data from the UCL corpus (Frank, Monsalve, et al., 2013). The eye-tracking tasks from this corpus consist of first-fixation time (duration of the first fixation on a word), first-pass time (summed duration of all fixations on a word before any fixation on a different word), right-bounded time (summed duration of all fixations on a word before any fixation on a subsequent word), and go-past time (summed duration of all fixations on a word and any preceding word up to the first fixation on a subsequent word). The first two measures are considered "early" measures which are influenced primarily by context-independent word-level features such as word frequency and word length, while the second two measures are considered "late" measures which are more influenced by context, and which measure visual attention (Rayner and Pollatsek, 2006). Visual attention is thought to strongly correlate with covert perceptual attention (Rayner, 2009). In addition to the UCL corpus, we include first-fixation, first-pass time, go-past time, total reading time (the summed duration of all fixations on a word), and word fixation count (the number of fixations on a word) from

the Ghent Eye-Tracking Corpus (GECO) (Cop et al., 2017). Finally, we include first-pass time, go-past time, and right-bounded time from the Dundee eye-tracking corpus (Kennedy, Hill, and Pynte, 2003). The train-test split for the UCL corpus is as described above in the paragraph about ERP tasks. The GECO and Dundee corpora are both split by taking the modulus of the initialization index over 4, giving 4 different train-test splits. These use 25% of the data for the validation set, and 20% of the remaining data as the task-sample-rate tuning set (see 5.3.4). The remainder becomes the primary training set. In all cases, the eye-tracking data is preprocessed in a manner similar to the ERP data, but with a log transform included. We first take the log of the data, then compute the mean and standard deviation on the content words in the training data within participant. We center and rescale the data such that for each eye-tracking measure and each participant, the mean of the measure over the content words in the training data is 0 and the standard deviation is 1. The data is then averaged over participants, and the mean and standard deviation are computed on these averaged values over the content words in the training data, and removed such that the averaged values have mean 0 and standard deviation 1 in the training data. The same process is applied to the held out sets, but the training means and standard deviations are used during both centering and rescaling steps. MSE is used as the loss for all eye-tracking data.

Self-paced reading time. Along with eye-tracking data, we include self-paced reading time from the UCL corpus (Frank, Monsalve, et al., 2013) as a behavioral task. In this version of self-paced reading, words from a sentence are shown one-by-one and the participant must press a button to advance to the next word in a sentence. The amount of time from the word presentation to the button press is recorded. The train-test split is as described for the UCL corpus in the ERP task paragraph above, and the data is preprocessed the same way as the UCL eye-tracking data. MSE is used as the loss for the self-paced reading time data.

C.2 More complete description of the MultiDDS algorithm and our variant of that algorithm

The intuition behind the MultiDDS algorithm proposed in Wang, Tsvetkov, and Neubig (2020) is that we want to learn from a task more frequently if it tends to move the shared model parameters in a direction which improves the loss for all tasks on unseen data. Wang, Tsvetkov, and Neubig (2020) starts by defining an objective function, J_{tune}, over all of the N tasks the model needs to learn on a set of data reserved for tuning how frequently the tasks are sampled (we refer to this dataset as "tune"). The tune set acts as our "unseen" data in the algorithm. We modify the notation from (Wang, Tsvetkov, and Neubig, 2020) slightly here:

$$J_{tune}(\gamma, \theta_1, D^1_{tune}, ..., \theta_N, D^N_{tune}) = \frac{1}{N}\sum_{i=1}^{N} J_i(\gamma, \theta_i, D^i_{tune})$$

Where $J_i(\gamma, \theta_i, D^i_{tune})$ is the loss function on task i parameterized by the parameters of the model that are common to all tasks, γ, the parameters of the model that are specific to task i, θ_i, and a tune dataset for task i, D^i_{tune}. This objective function minimizes the average (over tasks) of the risk on the tune set. Then, a probability distribution parameterized by ψ, $P_D(\psi)$, is defined such that the probability of sampling the ith task is given by:

$$P_D(\psi)_i = \frac{\exp \psi_i}{\sum_{k=1}^{N} \exp \psi_k}$$

We then aim to alternately optimize the model parameters, γ and θ_i, and the distribution parameters, ψ. When updating the model parameters, we optimize the model parameters for the expected loss given the

task-weighting distribution:

$$\gamma^*, \theta_1^*, ..., \theta_N^* = \text{argmin}_{\gamma,\theta_1,...,\theta_N} \mathbb{E}_{i \sim P_D(\psi)} J_i(\gamma, \theta_i, D_{tune}^i)$$

To update ψ, reinforcement learning is used with a reward function, $R(\gamma^{(t)}, i)$, where $\gamma^{(t)}$ is the value of γ at training step t, and i identifies a task. The reward function rewards tasks if the gradient with respect to the common model parameters γ on the training set is on average in the same direction as the gradient with respect to γ for all tasks on the tune set. In other words, a task i is rewarded if an update to γ in the direction given by the loss on task i is likely to benefit all tasks on the tune set (Wang, Tsvetkov, and Neubig, 2020):

$$\begin{aligned} R(\gamma^{(t)}, i) &\approx \nabla_\gamma J_{tune}(\gamma, \theta_i^{(t)}, D_{tune}^i, ..., \theta_N^{(t)}, D_{tune}^N)^\top \nabla_\gamma J_i(\gamma, \theta_i^{(t-1)}, D_{train}^i) \\ &= \cos\left(\nabla_\gamma \left(\frac{1}{N}\sum_{k=1}^{N} J_k(\gamma, \theta_k^{(t)}, D_{tune}^k)\right), \nabla_\gamma J_i(\gamma, \theta_i^{(t-1)}, D_{train}^i)\right) \end{aligned}$$

The reward function can be derived by differentiating J_{tune} with respect to γ (Wang, Tsvetkov, and Neubig, 2020). Using the REINFORCE algorithm (Williams, 1992), this reward function leads to the update rule (Wang, Tsvetkov, and Neubig, 2020):

$$\psi_i^{(t+1)} \leftarrow \psi_i^{(t)} R(\gamma^{(t)}, i) \cdot \nabla_{\psi_i} \log P_D(\psi)$$

Note that in practice we replace R by a scoring function, $S(\gamma^{(t)}, i)$, which only looks at a single batch b_{tune}^i or b_{train}^i, not the entire training or tuning set:

$$S(\gamma^{(t)}, i) = \cos\left(\nabla_\gamma \left(\frac{1}{N}\sum_{k=1}^{N} J_k(\gamma, \theta_k^{(t)}, b_{tune}^k)\right), \nabla_\gamma J_i(\gamma, \theta_i^{(t-1)}, b_{train}^i)\right)$$

Computing the scoring function $S(\gamma^{(t)}, i)$ for one task i requires a forward and backward pass through the model on the tune set for every task. Computing $S(\gamma^{(t)}, i)$ for all tasks therefore requires a number of forward and backward passes through the model which is quadratic in the number of tasks. Even though the MultiDDS algorithm only computes this scoring function every M steps, this is too slow in our setting of roughly 80 tasks if we want to update ψ regularly.

Our changes to MultiDDS. In our variant of the MultiDDS algorithm, rather than running a quadratic number of forward and backward passes through the model (with respect to the number of tasks) every M steps, we run one extra forward and backward pass for each of the M normal training steps we take. We define a matrix T to hold exponential moving averages of the cosine similarity between the gradients of tasks i and k. At step t, we take a standard training step by sampling a task i, and updating model parameters γ and θ_i. As part of the standard step, we compute the gradient $g_i = \nabla_\gamma J_i(\gamma, \theta_i^{(t-1)}, b_{train}^i)$. Then we run an additional forward and backward pass to compute a gradient on the tune set for a task k, $f_k = \nabla_\gamma J_k(\gamma, \theta_k^{(t)}, b_{tune}^k)$, and we update T with moving average parameter d:

$$T_{i,k} \leftarrow d \cdot T_{i,k} + (1 - d) \cdot \cos(g_i, f_k)$$

Then, after M standard training steps, we can estimate $R(\gamma^{(t)}, i)$, replacing any entries in T that have never been sampled by the mean of all entries in T that have been sampled. Letting $Q_{i,k} = 1$ if pair (i, k) has

been sampled and $Q_{i,k} = 0$ otherwise, and letting $\alpha = \frac{\sum_{i,k=1}^{N} Q_{i,k} \cdot T_{i,k}}{\sum_{i,k=1}^{N} Q_{i,k}}$:

$$R(\gamma^{(t)}, i) \approx \frac{1}{N} \sum_{k=1}^{N} Q_{i,k} \cdot T_{i,k} + (1 - Q_{i,k}) \cdot \alpha$$

This approximation requires only twice the number of standard forward and backward passes through the model during the coarse of training, and is independent of the number of tasks (though many pairs will not be sampled if the number of tasks is large). Critically, while tasks in the standard model update step are sampled according to $P_D(\psi)$, the tasks in the similarity computation step are sampled uniformly. While this does not make all pairs equally likely to be sampled, it does mean that for a task i that we sample in the training set, we are equally likely to update the cosine similarity with any task k in the tune set, so the score for task i still reflects (in expectation) the average of the similarities with all tasks. We also update the task specific parameters of the model in both the standard model update and in the similarity sampling step. This keeps all of the task specific model parameters tuned to the latest common model parameters throughout training even if a task is no longer sampled frequently to update the common model parameters. The common model parameters γ are only updated by the standard model update step. Finally, we make one other modification to standard MultiDDS. We express preferences for certain tasks, and we use a weighted average of the similarity of a task to other tasks when computing a task score, so if the preference for task k is given by p_k, and we denote the scoring function by $R_p(\gamma^{(t)}, i)$, our new approximation is:

$$R_p(\gamma^{(t)}, i) \approx \frac{\sum_{k=1}^{N} Q_{i,k} \cdot p_k \cdot T_{i,k} + (1 - Q_{i,k}) \cdot p_k \cdot \alpha}{\sum_{k=1}^{N} p_k}$$

The full training algorithm is given in algorithm 1. In summary the changes we make to the standard MultiDDS algorithm are:

- We approximate the scoring function in the original algorithm. The original computes, for each task pair, the cosine similarity of the gradient with respect to the common model parameters γ. This requires a quadratic number of forward in backward passes through the model. We instead sample one task pair for every standard model parameter update step, and keep an exponential moving average of gradient similarity between each task pair. Our modification requires only one extra forward and backward pass through the model for each standard forward and backward pass.
- We change the unweighted average of losses over tasks to a weighted average, which allows us to express a preference for some tasks.

We use a preference of 10 on the fMRI tasks and a preference of 1 on every other task. The probability of sampling a particular task after all six epochs of training (mean over all model initializations) is shown in figure D.12 along with the final similarity matrix T (mean over all model initializations). We also show the

variation in the sampling probability in figure D.13.

Algorithm 1: Training with modified MultiDDS

input : D_{train}; D_{tune}; M: number of steps between updating ψ, d: moving-average decay rate, p: vector of task preferences

output : A trained model

; // Initialize each ψ so that the probability of sampling the ith task is proportional to the relative size of the dataset for that that task

initialize $P_D(\psi)$ s.t. $P_D(\psi)_i = \frac{|D^i_{train}|}{\sum_{j=1}^{N}|D^j_{train}|}$, $\forall i$;

$step \leftarrow 1$;

initialize $Q \in \mathbb{R}^{N\times N} \leftarrow 0$;

initialize $T \in \mathbb{R}^{N\times N} \leftarrow 0$;

while *not converged* **do**

  sample task $i \sim P_D(\psi)$;

  sample batch b^i_{train} of items for task i from D_{train};

  $g_i \leftarrow \nabla_\gamma J(\gamma, \theta_i, b^i_{train})$;

  $\gamma \leftarrow \text{GradientUpdate}(\gamma, g_i)$;

  $\theta_i \leftarrow \text{GradientUpdate}(\theta_i, \nabla_{\theta_i} J(\gamma, \theta_i, b^i_{train}))$;

  sample task $k \sim Uni$;

  sample batch b^k_{tune} of items for task k from D_{tune};

  $\theta_k \leftarrow \text{GradientUpdate}(\theta_k, \nabla_{\theta_k} J(\gamma, \theta_k, b^k_{tune}))$;

  $T_{i,k} \leftarrow d \cdot T_{i,k} + (1-d) \cdot \cos(g_i, \nabla_\gamma J(\gamma, \theta_k, b^k_{tune}))$;

  $Q_{i,k} \leftarrow 1$;

  if $\text{modulo}(step, M) = 0$ **then**

    $\alpha \leftarrow \frac{\sum_{i,k=1}^{N} Q_{i,k}\cdot T_{i,k}}{\sum_{i,k=1}^{N} Q_{i,k}}$;

    $R^i_p \leftarrow \frac{\sum_{k=1}^{N} Q_{i,k}\cdot p_k \cdot T_{i,k} + (1-Q_{i,k})\cdot p_k \cdot \alpha}{\sum_{k=1}^{N} p_k}$, $\forall i$;

    $\psi \leftarrow \text{GradientUpdate}\left(\psi, -\nabla_\psi \left\{\sum_i R^i_p \cdot \log \frac{\exp(\psi_i)}{\sum_k \exp(\psi_k)}\right\}\right)$;

  end

  $step \leftarrow step + 1$;

end

C.3 Additional training and evaluation details

To train our model, we sample 1000 batches of training data in each of 6 training epochs using the variant of the MultiDDS (Wang, Tsvetkov, and Neubig, 2020) algorithm described in section 5.3.4. To sample a batch of data, we first sample a task according to the parameters of the task distribution, then we randomly choose 16 examples from that task for training. During evaluation, for each task having fewer than $500 \times 16 = 8000$ examples in the validation data, we make predictions for all examples. For each task having more than 8000 examples in the validation data, we sample 8000 examples without replacement in batches of 16 and make predictions for those examples (evaluating every example in all datasets takes too much time). For parameters that do not exist in the pretrained BERT model (i.e. the parameters of the functions which compute the latent space and the functions which compute our output predictions), we set the base learning rate to .001. For parameters that already exist in the pretrained BERT model, we set the base learning rate to .00001. We find that this combination of learning rates enables the model to learn without becoming

unstable. We ramp up the learning rate linearly during the first two epochs, and then apply exponential decay during the remainder of training. We update the parameters to the task sampling distribution every 100 steps. We also scale the fMRI losses up by a factor of 10, and we express a $10x$ preference for the fMRI tasks in the algorithm which adjusts the task sampling distribution. We find that a strong emphasis on the fMRI tasks is required to prevent the model from ignoring the noisy fMRI signal and to keep the shared representation in a part of the space that is useful for both fMRI prediction and NLP task prediction. We apply a weight decay penalty of 0.01 to the weights in all of the layers of the model including the output weights. The code is available at https://github.com/danrsc/bert_brain.

Appendix D

Additional Results for the Model Parameter Similarity Method

D.1 Accuracies and correlations for chapter 5

Although model prediction accuracy is not the primary focus of our work, in order for a comparison of the weights involved in two task predictions to be meaningful, the predictions for both tasks must be good, which we define in our work as being statistically significantly better than a baseline. For regression tasks, we use Pearson's correlation between predicted and actual task values as our metric, and we use 0 correlation as the baseline. For classification tasks, we use accuracy as our metric, and we use the accuracy that is achieved by predicting the most common value (mode accuracy) in the validation data as our baseline. We compute p-values by using the 100 values of the appropriate metric (either correlation or accuracy) across the 100 model initializations. For regression tasks, we use a 1-sample t-test to test for a correlation greater than 0, and for classification tasks, we use a 2-sample related t-test (the model accuracy and baseline (mode) accuracy both change for each model initialization because the validation set accuracies are computed by sampling the data in the validation set). Our significance threshold is a p-value of 0.01, after false-discovery rate correction for multiple comparisons using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001). When applying false-discovery rate correction, we group together the data which we expect is approximately from the same distribution. Thus the eye-tracking data is all grouped together, the event related potential (ERP) data is all grouped together, and each fMRI participant's data is its own group. All other p-values are used without correction. For all tasks apart from fMRI prediction, the accuracies and correlations can be found in table D.1. Figures D.1 and D.2 show a summary and the spatial locations respectively of the correlation between predicated and actual values for fMRI voxels.

D.2 Additional observations about NLP similarities not included in chapter 5

Competitions between arguments observed not just in ARG0 prediction. In some cases where predictions for an NLP task rely on two spans, we see the same symmetry between the spans as we see for the prediction of the ARG0 class (discussed in detail in chapter 5), i.e. one span seems to partially negate the other. These suggest that the model uses competition between arguments for predictions other than just for ARG0. The manipulated-by-another prediction has a similar symmetry (figure D.4) — and, like ARG0, it has that property that only one argument in a clause tends to have this label. Some dependency role labels also show this symmetry (figure D.8, note that in the case of dependency roles, span 0 is the dependent and span 1 is the context). The nsubj (subject of a sentence, e.g. ***Books*** *are fun to read*) dependency role shows this symmetry, which is not suriprising considering that the subject task is highly similar to the ARG0 task. The object dependency role also shows the symmetry, and the oblique dependency role exhibits it to a lesser extent (we note that multiple obliques can appear in a sentence). A different kind of competition can also be seen in the dependency role span 1 (context) profiles — we see that objects are less likely in the context of obliques and nominal modifiers, and obliques are less likely in the context of nominal modifiers. In general, we hypothesize that the context embedding is used for cross-talk between arguments when the presence of one argument influences the likelihood of another.

D.2.1 Word-level and span-level similarity profiles

ARG1 and ARG2. In figure D.3, we see that, in contrast to ARG0, ARG1 appears to be primarily predicted using syntactic rather than semantic features. The similarity profile of the argument span for ARG1 (sr.ARG1.span1) essentially suggests that the model is looking for a noun or noun phrase which is not the governor in a complement or subclause. This fits quite well with the definition of ARG1, and we

Task	Acc. / Corr.	Mode Acc.	Task	Acc. / Corr.	Mode Acc.
elan	**0.43**	n/a	**pr1.makes_phys_contact**	**0.88**	0.86
epnp	**0.39**	n/a	**pr1.manip_by**	**0.82**	0.57
lan	**0.45**	n/a	**pr1.pred_changed_arg**	**0.71**	0.63
n400	**0.37**	n/a	**pr1.sentient**	**0.94**	0.77
p600	**0.43**	n/a	**pr1.stationary**	**0.95**	0.95
pnp	**0.45**	n/a	**pr1.volition**	**0.92**	0.67
Dundee first-pass	**0.41**	n/a	**pr2.awareness**	**0.87**	0.56
Dundee go-past	**0.39**	n/a	pr2.change_of_loc	0.81	0.82
Dundee right-bounded	**0.44**	n/a	**pr2.change_of_possess**	**0.92**	0.92
GECO first-fixation	**0.20**	n/a	**pr2.change_of_state**	**0.71**	0.70
GECO fixation-count	**0.73**	n/a	**pr2.chng_of_stat_cont**	**0.58**	0.53
GECO first-pass	**0.41**	n/a	pr2.changes_possess	1.00	1.00
GECO go-past	**0.41**	n/a	pr2.existed_after	0.88	0.89
GECO total reading time	**0.32**	n/a	pr2.existed_before	0.82	0.83
UCL first-fixation	**0.72**	n/a	pr2.existed_during	0.98	0.98
UCL first-pass	**0.72**	n/a	pr2.exists_as_physical	0.98	0.98
UCL go-past	**0.72**	n/a	**pr2.instigation**	**0.68**	0.66
UCL right-bounded	**0.72**	n/a	pr2.loc_of_event	0.99	0.99
bshift	**0.60**	0.52	pr2.makes_phys_contact	1.00	1.00
constituent	**0.54**	0.37	pr2.partitive	0.78	0.79
coordinate-inversion	0.52	0.52	pr2.pred_changed_arg	1.00	1.00
coreference	**0.80**	0.78	**pr2.sentient**	**0.86**	0.52
dependency-role	**0.60**	0.12	pr2.stationary	1.00	1.00
definite-pronoun resolution	0.50	0.50	**pr2.volition**	**0.85**	0.53
named-entity	**0.79**	0.20	**pr2.was_for_benefit**	**0.62**	0.57
object-number	**0.59**	0.52	pr2.was_used	0.84	0.84
part-of-speech	**0.62**	0.13	**self-paced reading time**	**0.17**	n/a
pr1.awareness	**0.91**	0.63	**SemEval**	**0.53**	0.16
pr1.change_of_loc	0.95	0.95	**sentence length**	**0.43**	0.19
pr1.change_of_state	**0.71**	0.63	**sentiment**	**0.64**	0.51
pr1.changes_possess	0.93	0.93	semantic odd man out	0.51	0.52
pr1.existed_after	**0.76**	0.69	**semantic role**	**0.59**	0.35
pr1.existed_before	**0.80**	0.65	**subject number**	**0.64**	0.52
pr1.existed_during	**0.86**	0.82	**tense**	**0.76**	0.52
pr1.exists_as_physical	**0.87**	0.66	**top constituents**	**0.31**	0.07
pr1.instigation	**0.88**	0.65	**tree depth**	**0.20**	0.19
pr1.loc_of_event	**0.93**	0.92			

Table D.1: Accuracies and correlations of the model (mean over 100 model initializations). Task names and performance numbers are bolded if they are significantly greater than the baseline with a (sometimes false-discovery rate corrected, see text) p-value < 0.01. Mode accuracy is used as the baseline and reported for classification tasks. For regression tasks, 0 is used as the baseline and n/a is listed in the table under mode accuracy.

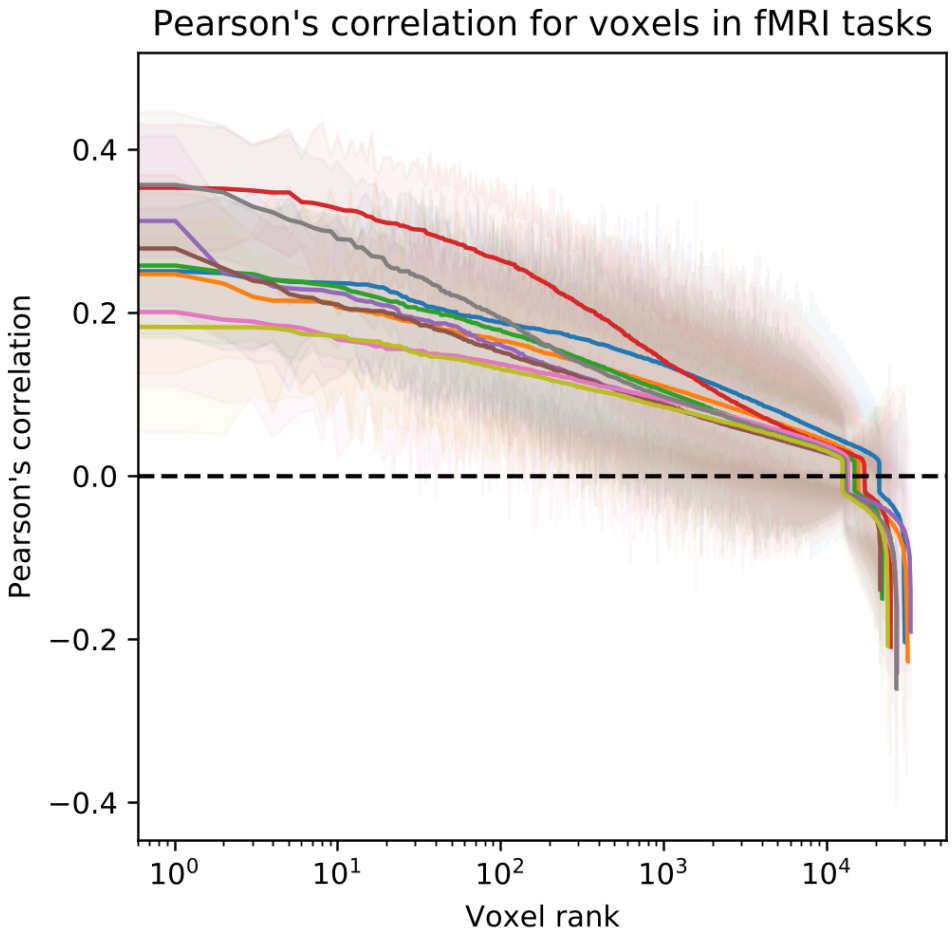


Figure D.1: Correlation between predicted and actual voxel values for all voxels where correlations were significantly different from 0 at a 0.01 level (significance computed using a single-sample t-test over 100 model initializations, and corrected for multiple comparisons within participant using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001). Voxels are sorted on the x-axis in order of descending correlation. Each color shows the correlations for one participant, and one standard deviation above and below each line is rendered in the lightly colored regions in the background.

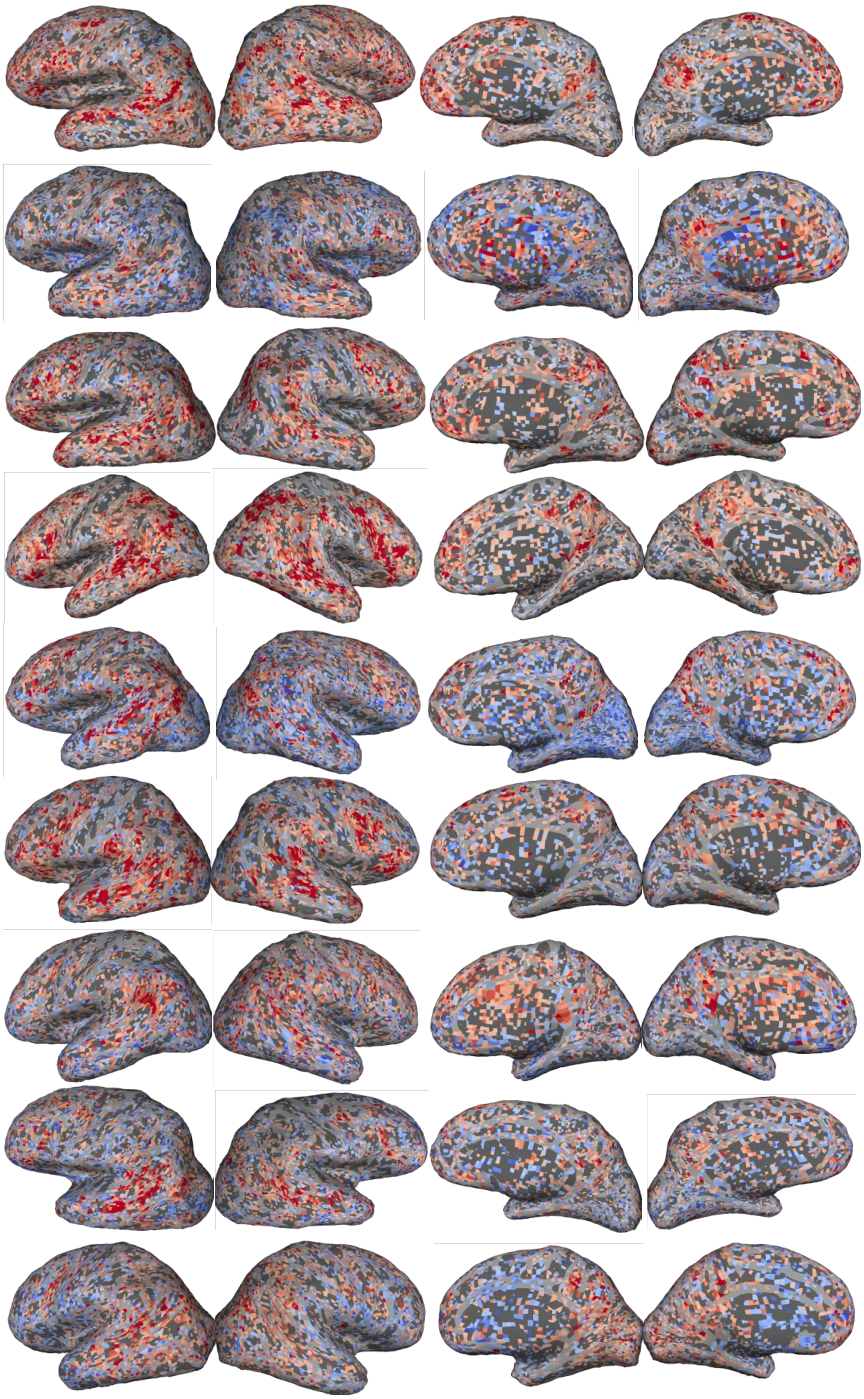


Figure D.2: Correlation between predicted and actual voxel values for all voxels where correlations were significantly different from 0 at a 0.01 level (significance computed using a single-sample t-test over 100 model initializations, and corrected for multiple comparisons within participant using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001). Each row shows the correlations for one participant. From left to right, columns show inflated left and right lateral views and right and left medial views of each participant's brain. Color indicates the value of the correlations. Colors are set independently for each participant such that the reddest values indicate the $95th$ percentile of the absolute value of the correlations between model predictions and actual values for that participant, and the bluest values are the negative of the reddest values.

also note the similarity to the dp.obj (object) dependency role, which is the syntactic counterpart to the ARG1 semantic role. The similarity profile of the verb context for ARG1 (sr.ARG1.span0) shows that if the predicate span is a verb phrase, that increases the likelihood of the ARG1 label, while if the predicate has similarities to several other syntactic constructions, the probability of ARG1 is decreased. The model does not appear to use competition for ARG1 as we hypothesize it does for ARG0 — note that the similarities in ARG1 are not negated between span 0 and span 1 in contrast to ARG0. Turning to ARG2, we see that ARG2 is predicted using semantic features, and indeed that its similarity profiles look much more like the proto-patient role from White, Rawlins, and Van Durme (2017) than the similarity profiles of ARG1. The ARG2 argument span (sr.ARG2.span1) profile nicely matches the proto-patient profile. The ARG2 verb-context span (sr.ARG2.span0) has agent-like characteristics in contrast to the verb-context span for ARG0 (and indeed the two are anti-similar to each other) because it is more likely to be present when a different argument is acting as ARG0. The similarity profile also suggests that when predicting ARG2, the model is mostly matching copulas (e.g. *John is a good dancer*) and prepositional phrases. It's similarity to personal pronoun prediction and anti-similarity to eye-tracking measures could be explained by the model relying in part on short word length to identify linking verbs.

Proto-role similarity profiles. The proto-role property tasks are highly similar to each other, especially the agent-like properties (awareness, volition, sentient, instigation). Agent-like properties are also highly similar to the agent of the SemEval Instrument-Agency relation and the Producer of the SemEval Product-Producer relation, while being anti-similar to the effect in the Cause-Effect relation. None of these similarities are surprising, and in some sense the similarity profiles "ground-out" in these proto-role properties since they are by definition decompositions of less granular semantic information. We do note some interesting aspects of these profiles. Like the ARG0 competition described above, the manipulated-by-another (pr1.man.span0, pr1.man.span1) profiles suggest a competition among arguments to a verb to be the strongest candidate for the manipulated-by-another prediction. We also note the prevalence of named entity classes as being either similar or anti-similar to physical/state properties (change of state, existed after, exists as physical, and predicate changed argument). The PERCENT class seems to act as a standin for a non-physical entity (which is borne out by its profile in figure D.6).

SemEval physicality, and similarities between SemEval and named entities. Turning to figure D.5, which shows similarity profiles for one direction of each relation, we note some differences between the SemEval relations and semantic roles. Semantic roles tend to be distinguished more by animacy attributes like awareness, volition, sentient, and instigation. Those are important for SemEval relation direction, but physicality (exists-before, exists-during, exists-after, makes-physical-contact, exists-as-physical, and change-of-state) is very important for distinguishing between relation types. Finally, we note that named entity tasks are also, in general, similar to relation argument tasks. In normal text, the arguments to a SemEval relation would often be named entities. Given the similarity between named entity classes and SemEval relation arguments, it's not surprising to see that named entity classes also show strong similarity to semantic proto-role properties. PERSON, NORP (nationality or religious or political group), and ORG show strong agent-like properties, while GPE (geo-political entity) is more weakly agent-like. DATE, MONEY, CARDINAL, ORDINAL, PERCENT, and TIME all show anti-physicality (all are anti-similar to existance properties), with DATE and MONEY also matching well with a proto-patient profile. In contrast, LOC is strongly physical. Along with the semantic decomposition properties of these entities, we also notice the model using syntactic clues in its predictions. Additionally, we see that number-related entities CARDINAL, MONEY, ORDINAL, and PERCENT are similar to each other, and DATE and TIME are similar to each other.

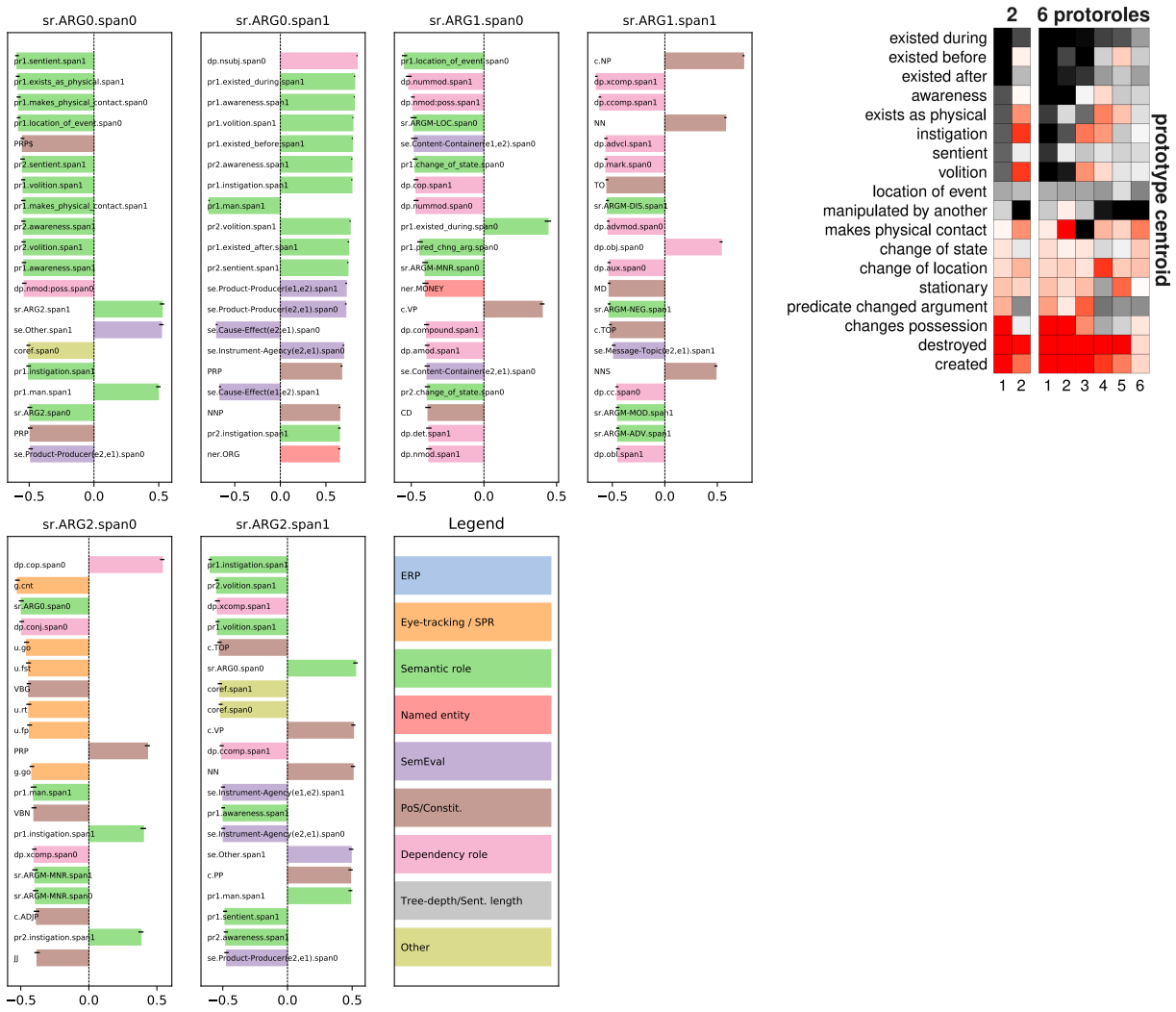


Figure D.3: Left: Each subplot shows the twenty largest magnitude task similarities for each span of three semantic role labels — ARG0, ARG1, and ARG2. Bar colors indicate the broad category of similar tasks as indicated by the legend. Right: Figure 4 from (White, Rawlins, and Van Durme, 2017) shows how properties relate to the latent predicate-general roles which they use as a modeling intermediary for mapping tokens to predicate-specific roles (PropBank roles) for either 2 latent roles (left of subplot) or 6 latent roles (right of subplot). In the heatmap, black indicates higher likelihood and red indicates lower likelihood. The proto-agent role as described by Dowty (1991) matches the leftmost column of both subplots. In the right subplot with 6 roles, the proto-agent has been split into two subcategories (animate and inanimate) (White, Rawlins, and Van Durme, 2017). All other columns in the figure are described in White, Rawlins, and Van Durme (2017) as non-agent columns or perhaps different kinds of proto-patient. The ARG0 argument span (sr.ARG0.span1) matches the proto-agent well and the ARG2 argument span (sr.ARG2.span1) matches the proto-patient well. The model's identification of ARG1 appears to rely more on syntactic clues. Surprisingly, the verb context span for ARG0 (sr.ARG0.span0) also appears to match the proto-patient profile, while the verb context span for ARG2 (sr.ARG2.span0) appears to have agentive characteristics. The reasons for this are discussed in detail in chapter 5.

Figure D.4: The 20 largest magnitude task similarities for selected proto-role properties. See text for discussion.

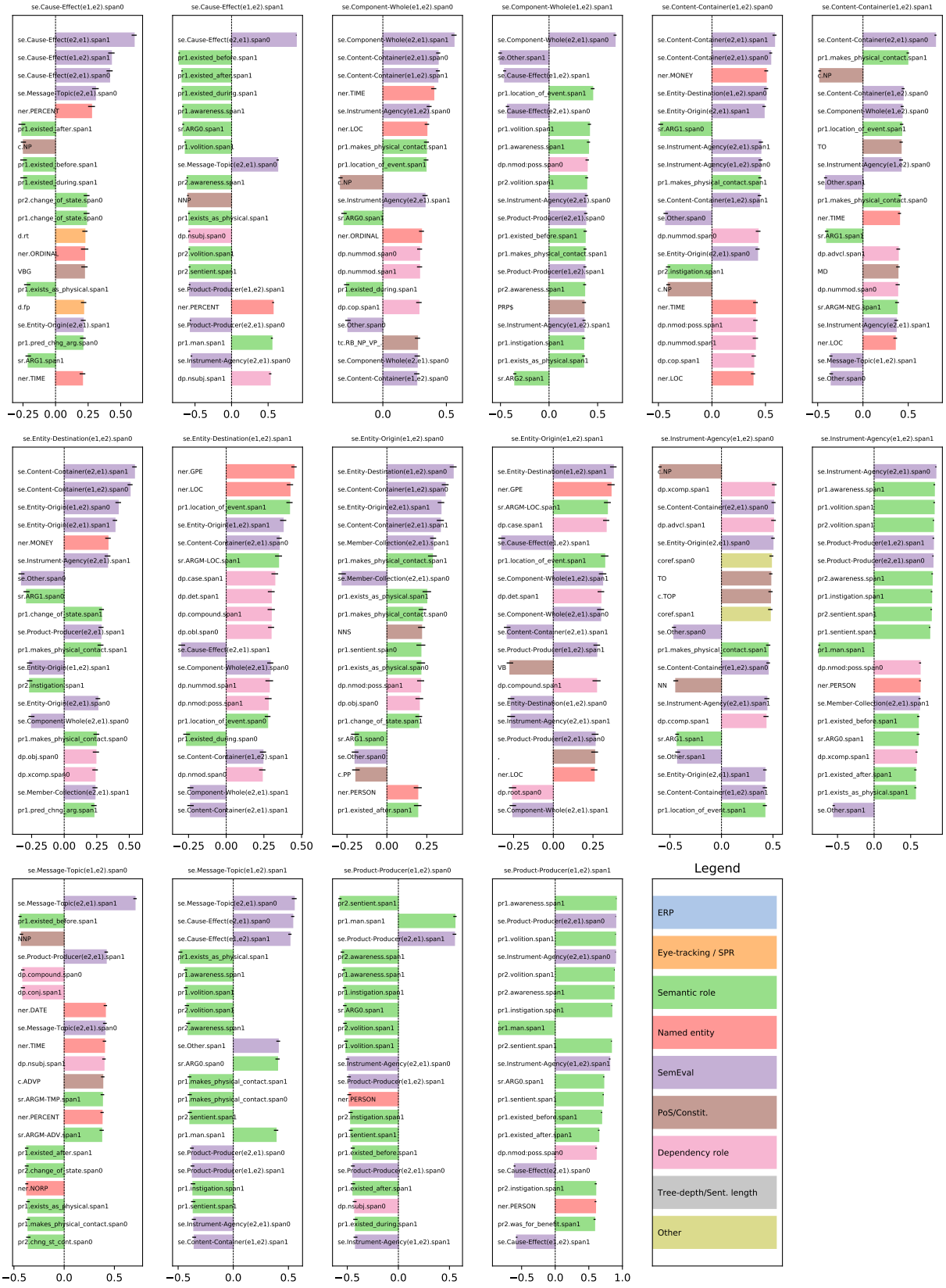


Figure D.5: The 20 largest magnitude task similarities for selected SemEval relations. See text for discussion.

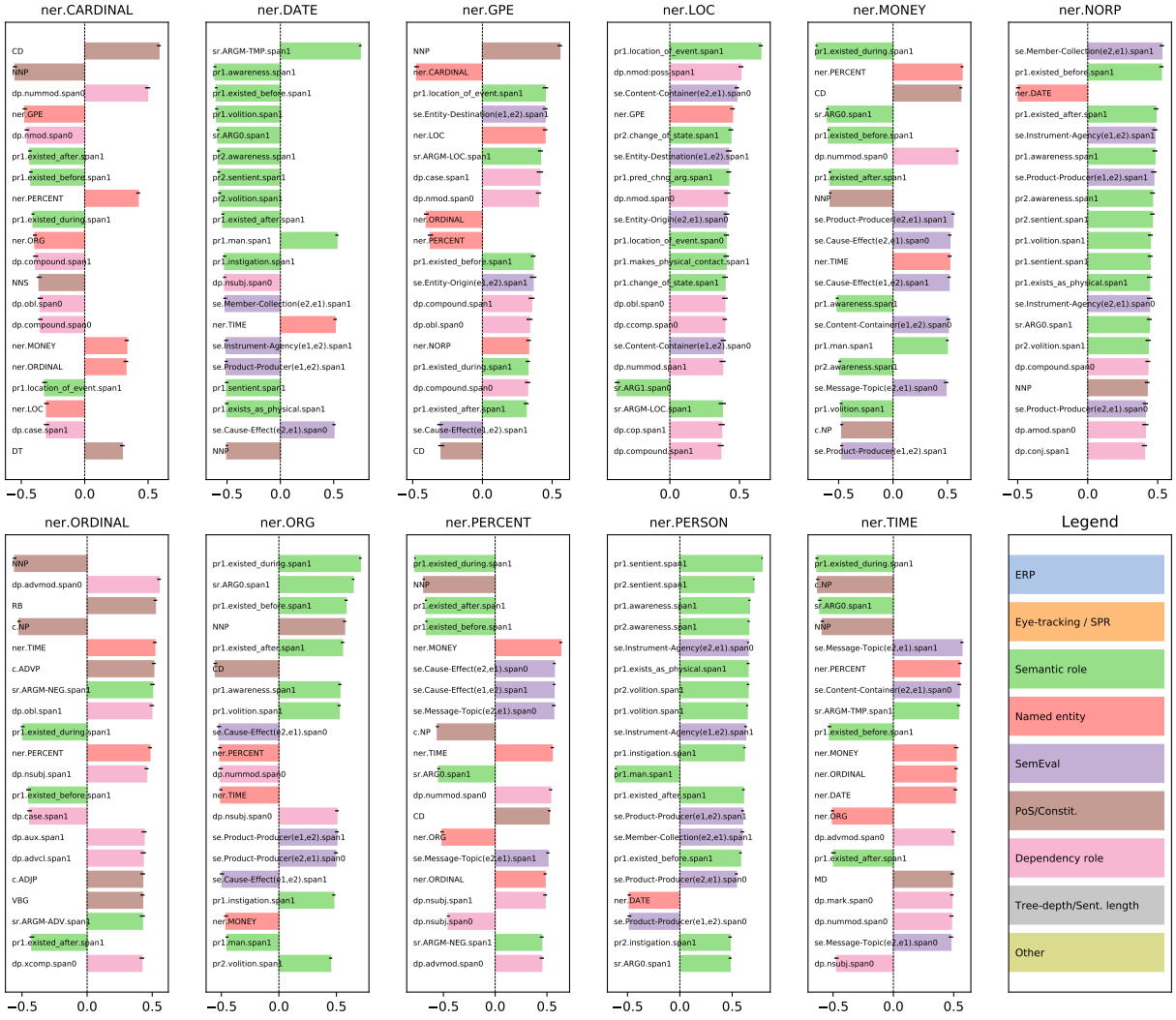


Figure D.6: The 20 largest magnitude task similarities for each of the predictable named entity classes which appear in at least 1% of examples. Note the agent-like properties of PERSON, NORP, ORG, and GPE, and the anti-physicality properties of DATE, MONEY, CARDINAL, ORDINAL, PERCENT, and TIME. In constrast, LOC is strongly physical. Number-related entities (CARDINAL, MONEY, ORDINAL, and PERCENT are similar to each other, and DATE and TIME are similar to each other.

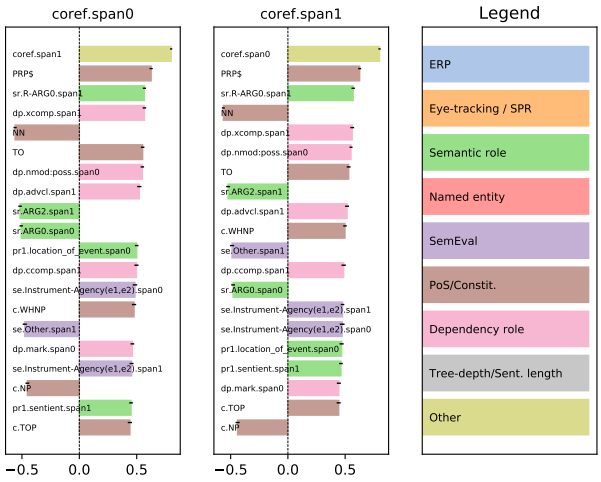


Figure D.7: The 20 largest magnitude task similarities for coreference prediction. See text for discussion.

Syntactic tasks are primarily similar to other syntactic tasks, with intuitive exceptions. In the dependency role task, a syntactic relation takes two arguments — the dependent and the governor/head. The dependent acts on the governor in some way. For example, in the nsubj relation, the dependent is the nominal subject and the governor is the head of the clause which the dependent is the subject of. In the sentence *Harry goes to the store.*, *Harry* would be the dependent of the nsubj relation while *goes* would be the governor. In our model, span 0 represents the dependent, and span 1 represents the governor. Figure D.8 shows the similarity profiles of selected dependency roles (for the full set, see figures D.20-D.22). We make several observations about these profiles. First, the dependency role tasks are largely similar to other dependency role tasks and to part-of-speech/constituency prediction, suggesting that dependency role prediction mostly relies on syntactic features. There are some notable exceptions: the dependent (span 0) in nmod:poss (a nominal modifier which is possessive, e.g. ***Marie**'s book*) is unsurprisingly strongly agent-like, as is the dependent in the nsubj relation, while the dependent in the obj (object) and obl (oblique) relations prefers to be inanimate and, for obj, patient-like. We also note two cognition-relevant points of interest. The adjective (span 0) in the amod relation is similar to prediction of eye-tracking data. This suggests that the model predicts that adjectives slow down reading (i.e. a higher probability of an adjective correlates with a higher value for eye-tracking measures such as first-pass time) — an intuitive result which one can hypothesize is the result of slower integration of meaning when adjectives are present or the result of increased eye-movements between words in a phrase. We also see similarity between the noun (span 0) in the oblique relation and the P600 ERP response, suggesting that the model predicts an increased P600 when a non-core argument is present in a sentence. This could be a consequence of timing, i.e. the relative timing of the N400 and P600 might be shifted during processing of an oblique, or it could reflect more attentive processing during the processing of an oblique (which the literature suggests would lead to an increased P600, see chapter 2). Alternatively, since we observe that the ERP component tasks are all similar to each other (see figure 5.7), this could just be an indicator of increased cognitive load. We also see that the determiner (span 0 of the determiner relation, see figure D.21) is anti-similar to ERP component tasks, which lends some credence to the cognitive load hypothesis. Part-of-speech and constituency task similarity profiles mostly follow the same template as dependency role similarity profiles. These can be found in figures D.17-D.18.

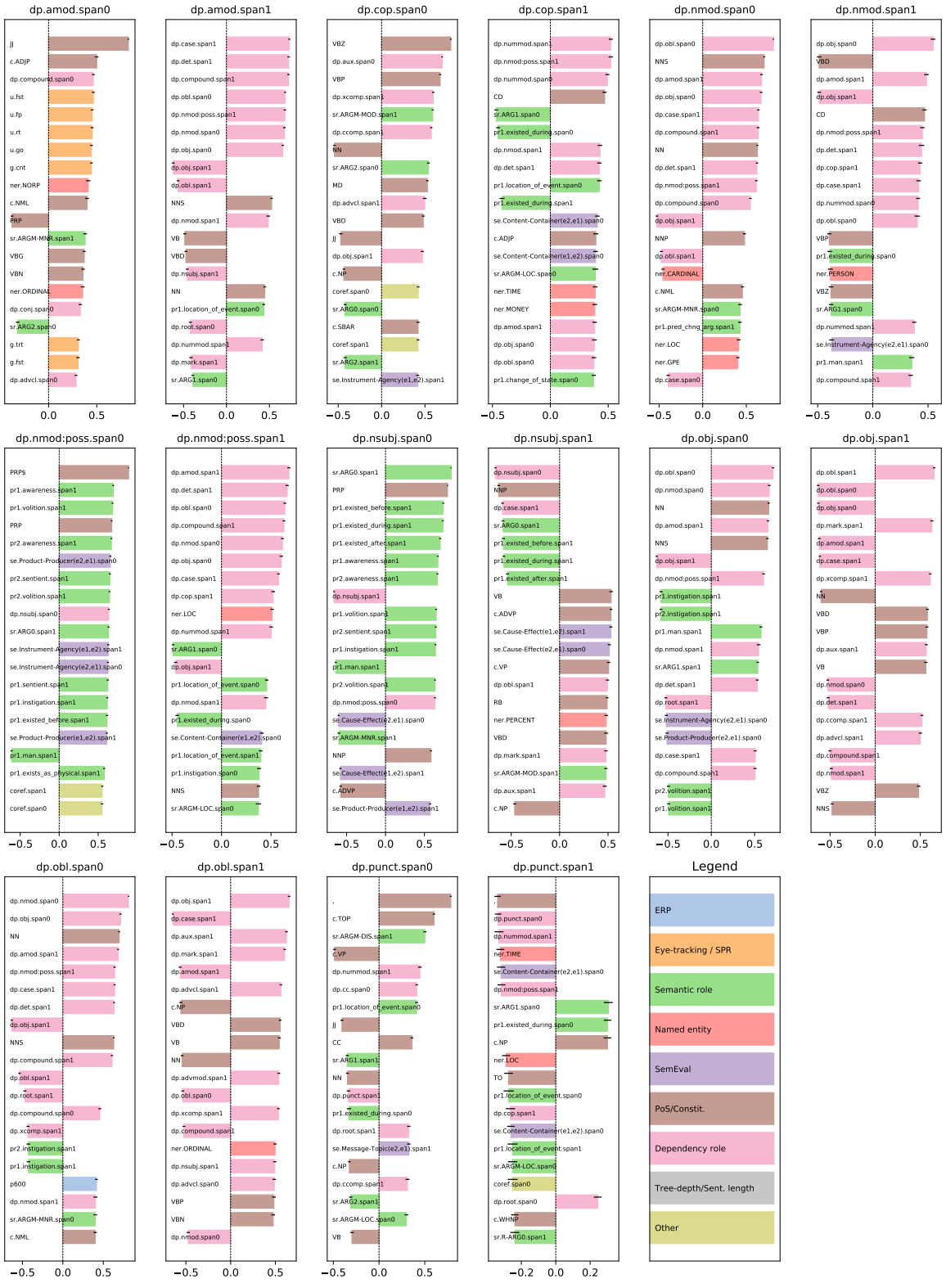


Figure D.8: Similarity profiles of selected dependency roles. See text for discussion.

D.2.2 Sequence level task similarities

Top constituents. The top constituents task frames constituent prediction as a sequence-level, 20-way classification task. The task is to predict the sequence of constituents immediately below the root S-node in the parse tree. For example, one sequence is NP-VP (a noun phrase followed by a verb phrase). This sequence, along with the eighteen other sequences that are observed most commonly in natural sentences plus an additional "Other" class comprise the twenty labels from which the model must choose a label for a sentence in this task. An immediate question of interest is what kind of information is used in this aggregate constituent prediction task. Figure D.9 shows the similarity profiles for the top-constituent classes. These are similar to each other, and also, sensibly, similar to sentence-length and tree-depth prediction tasks. Additionally, we see that many top constituent tasks have similarities to both semantic tasks (proto-role properties, named entities, SemEval relations, and semantic roles) and syntactic tasks (dependency roles, constituents, and part of speech). We do not typically, however, see the individual constituents which are a part of the constituent sequence being predicted (for example, for the tc.NP_VP_. task, we do not see similarities to the c.NP or c.VP constituent tasks, but we do see similarity to PRP$ (possessive pronoun) prediction). In short, it appears that both syntactic and semantic information about individual words and their relationships in the sequence is retained in the sequence level representation, and that this information, along with information about the sentence length and tree depth, is used to guide the top-constituents prediction.

Binary sequence-level tasks. Binary sequence-level task similarity profiles are shown in D.10. Surprisingly, sentiment classification appears to be essentially a proto-agent profile, albeit with reduced similarity values. We do not have a good hypothesis for this result, other than that this heuristic might work in the corpus — perhaps reviewers are more likely to use passive voice when ratings are negative. Bigram shift (whether two random words have been swapped) is anti-similar to eye-tracking data, which as we discuss in chapter 5 in the section about eye-tracking, is likely related to word length and the presence of multi-word expressions. This suggests that the model may be making its prediction based on absence of multi-word expressions (due to the swapping of words breaking up those expression), though we do not directly observe anti-similarity to multi-word expression tasks. Object number and subject number (whether the object and subject respectively are plural) are interesting. Both are more likely to be plural if existed_during is true (i.e. both are more likely if the sequence token has features indicating physicality). Both also make use of features similar to those used by SemEval tasks to determine whether the arguments are involved in a Content-Container, Instrument-Agency, Component-Whole, or Product-Producer relation. All of those decrease the probability that the object or subject is plural. Subject number also makes use of whether there is a determiner or a numerical quantity in the sentence. All of these suggest that a surprising amount of information about the individual tokens in the sentence is embedded in the sequence level representation. Finally, the tense task (whether the verb is past tense) is very similar to top-constituent prediction. It makes use of tree-depth and sentence-length information, and is less likely to predict past tense in the presence of physicality or state change.

D.3 Infreqeuent classes

In figure D.11 we show the similarity matrix for all of the semantic role classes irrespective of how frequently they appear in the examples in the dataset. This illustrates the point, which we observe across multiple tasks, that the model appears to learn a single representation for all infrequently occurring classes rather than modeling each separately. Since these representations are uninformative, we remove these

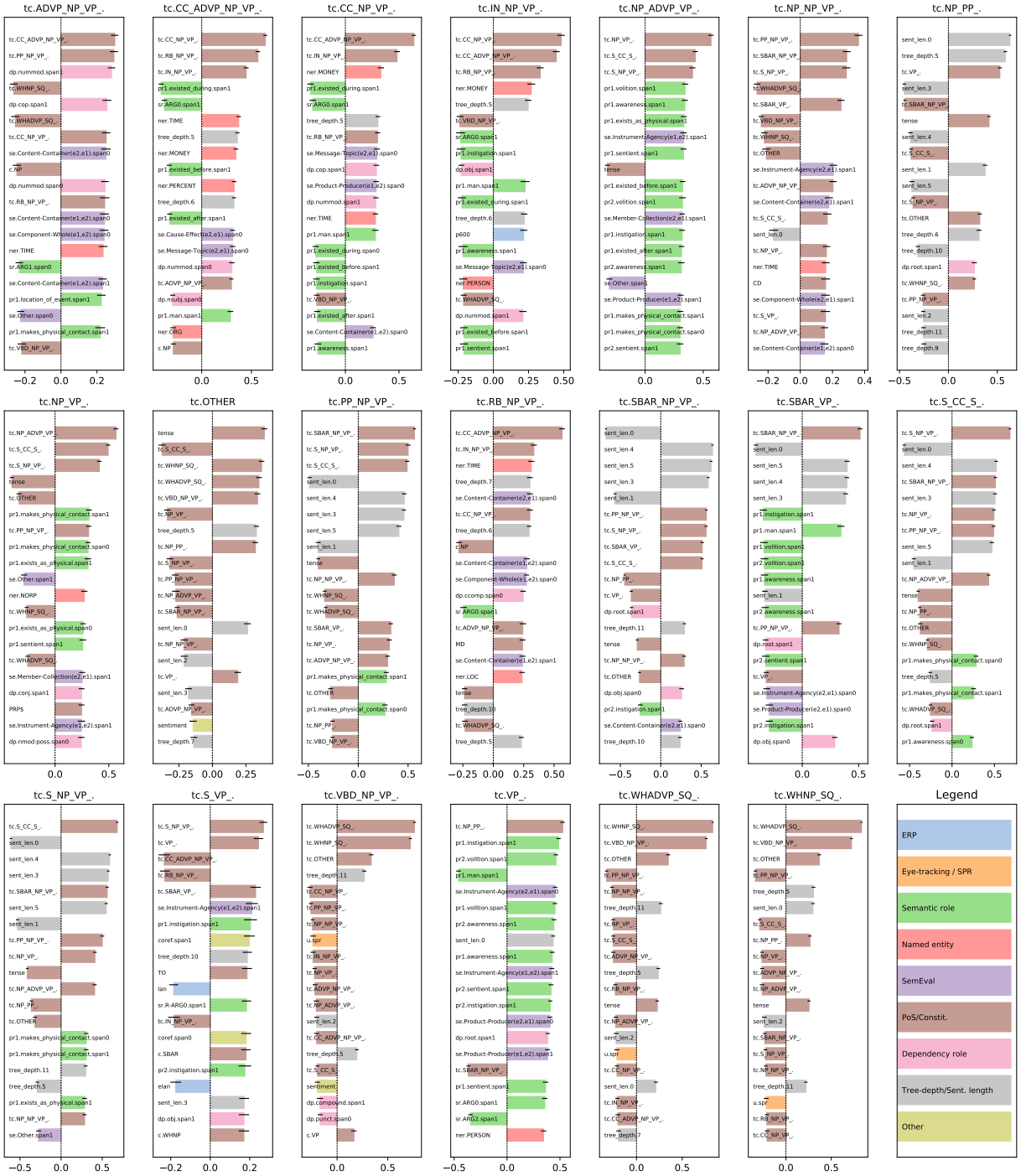


Figure D.9: Similarity profiles of top-constituency classes. See section 5.4.2 for discussion.

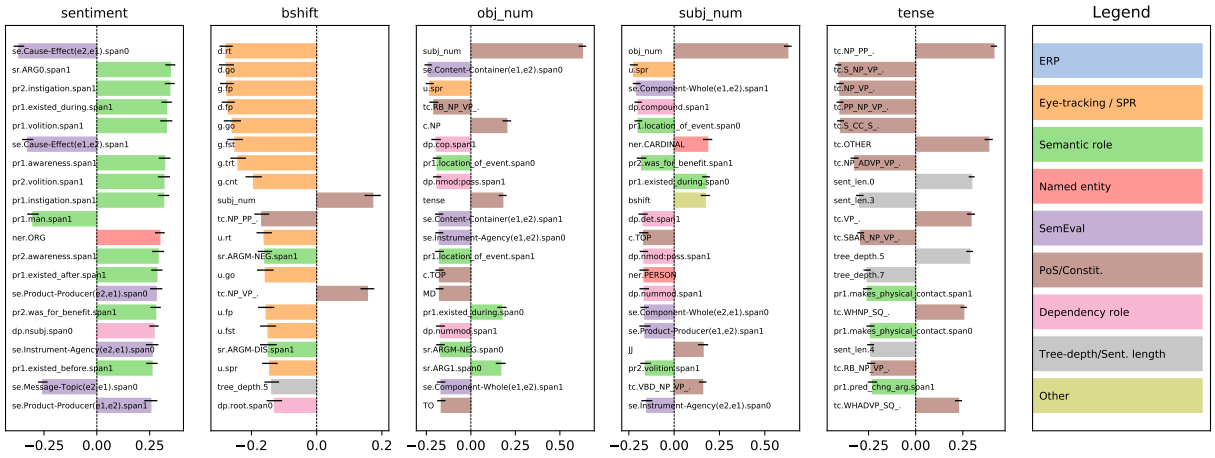


Figure D.10: Similarity profiles of sequence-level binary tasks. See text for discussion.

infrequent classes from our similarity profiles.

D.4 Task sampling probabilities

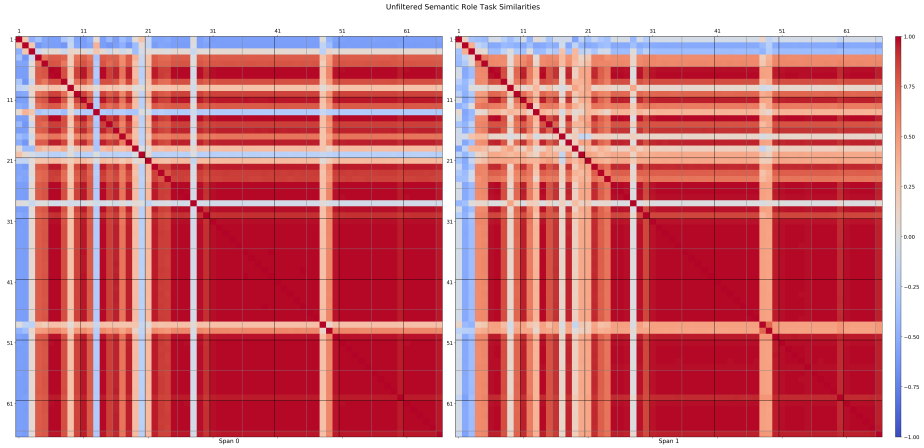
Figure D.12 shows the mean over model initializations of the sampling probability for each task at the end of training after running the variant of the MultiDDS algorithm described in chapter 5 along with the gradient similarities that our variant of the MultiDDS algorithm estimates for all tasks. Figure D.13 shows how the task sampling distribution looks for each model initialization to illustrate how it varies.

D.5 Full similarity results for chapter 5

Figures D.14 and D.15 show the complete similarity matrix for fMRI tasks compared to all non-fMRI tasks and for all non-fMRI tasks compared to each other (all tasks/classes in these matrices meet the predicatability and frequency criteria described in chapter 5). The remainder of the figures in this section show all of the similarity profiles for all of the tasks not discussed elsewhere.

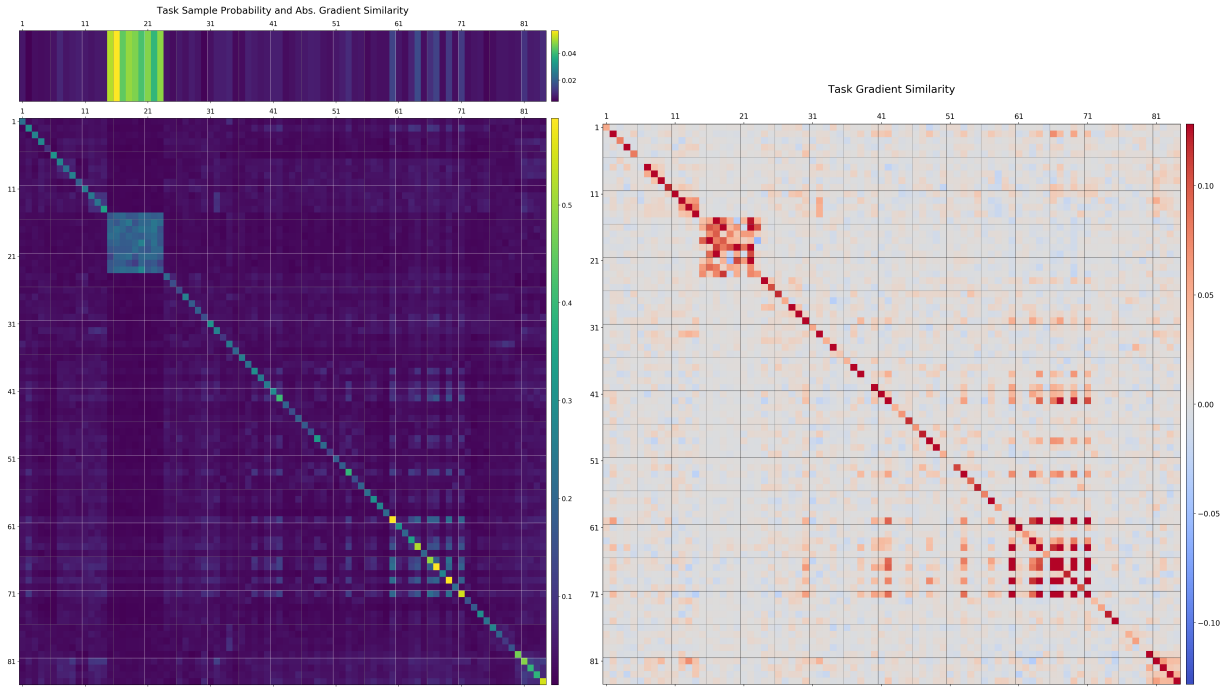
D.6 fMRI summaries

The figures in this section show the fMRI summary similarity profiles and absolute similarity profiles as computed in 5.3.4 for participants not shown in chapter 5.



1. **ARG0**
2. **ARG1**
3. **ARG2**
4. ARG3
5. ARG4
6. ARG5
7. ARGA
8. ARGM-ADJ
9. **ARGM-ADV**
10. ARGM-CAU
11. ARGM-COM
12. ARGM-DIR
13. **ARGM-DIS**
14. ARGM-DSP
15. ARGM-EXT
16. ARGM-GOL
17. **ARGM-LOC**
18. ARGM-LVB
19. **ARGM-MNR**
20. **ARGM-MOD**
21. **ARGM-NEG**
22. ARGM-PNC
23. ARGM-PRD
24. ARGM-PRP
25. ARGM-PRR
26. ARGM-PRX
27. ARGM-REC
28. **ARGM-TMP**
29. C-ARG0
30. C-ARG1
31. C-ARG2
32. C-ARG3
33. C-ARG4
34. C-ARGM-ADJ
35. C-ARGM-ADV
36. C-ARGM-CAU
37. C-ARGM-COM
38. C-ARGM-DIR
39. C-ARGM-DIS
40. C-ARGM-DSP
41. C-ARGM-EXT
42. C-ARGM-LOC
43. C-ARGM-MNR
44. C-ARGM-MOD
45. C-ARGM-NEG
46. C-ARGM-PRP
47. C-ARGM-TMP
48. **R-ARG0**
49. R-ARG1
50. R-ARG2
51. R-ARG3
52. R-ARG4
53. R-ARG5
54. R-ARGM-ADV
55. R-ARGM-CAU
56. R-ARGM-COM
57. R-ARGM-DIR
58. R-ARGM-EXT
59. R-ARGM-GOL
60. R-ARGM-LOC
61. R-ARGM-MNR
62. R-ARGM-MOD
63. R-ARGM-PNC
64. R-ARGM-PRD
65. R-ARGM-PRP
66. R-ARGM-TMP

Figure D.11: Unfiltered similarities of semantic role classes to each other, with span 0 shown in the left subplot and span 1 shown in the right subplot for clarity. The enumerated list on the right shows the order in which the classes are rendered. Bold indicates that a class appears in at least 1% of the examples. The classes which are nearly identical to each other appear in fewer than 1% of examples, and we filter these out in other results.



1. bshift
2. constituency
3. coord-inv
4. coref
5. dep. role
6. def. pron. res.
7. dundee.first_pass
8. dundee.go_past
9. dundee.rt_bnd.
10. elan
11. epnp
12. ucl.first_fix.
13. ucl.first_pass
14. ucl.go_past
15. fmri_F
16. fmri_G
17. fmri_H
18. fmri_I
19. fmri_J
20. fmri_K
21. fmri_L
22. fmri_M
23. fmri_N
24. lan
25. n400
26. ner
27. obj_num
28. p600
29. pnp
30. part of speech
31. ucl.self_paced
32. ucl.rt_bnd.
33. sem_eval
34. sent_len
35. sentiment
36. somo
37. pr1.awareness
38. pr1.chg_of_loc
39. pr1.chg_of_state
40. pr1.changes_poss
41. pr1.created
42. pr1.destroyed
43. pr1.existed_after
44. pr1.existed_before
45. pr1.existed_during
46. pr1.exists_as_phys
47. pr1.instigation
48. pr1.loc_of_event
49. pr1.makes_phys_contact
50. pr1.manip_by
51. pr1.pred_chg_arg
52. pr1.sentient
53. pr1.stationary
54. pr1.volition
55. pr2.awareness
56. pr2.chg_of_loc
57. pr2.chg_of_poss
58. pr2.chg_of_state
59. pr2.chg_of_state_cont
60. pr2.chg_poss
61. pr2.existed_after
62. pr2.existed_before
63. pr2.existed_during
64. pr2.exists_as_physical
65. pr2.instigation
66. pr2.location_of_event
67. pr2.makes_phys_contact
68. pr2.partitive
69. pr2.pred_chg_arg
70. pr2.sentient
71. pr2.stationary
72. pr2.volition
73. pr2.was_for_benefit
74. pr2.was_used
75. srl
76. subj_num
77. tense
78. top_const
79. tree_depth
80. geco.first_fix.
81. geco.fix_cnt.
82. geco.first_pass
83. geco.go_past
84. geco.tot_rd_time.

Figure D.12: Mean over model initializations of the final state of the task sampling algorithm. **Left top:** the probability of sampling a task. **Left bottom:** the mean of the absolute value of the similarities between the gradients of tasks with respect to the shared parameters. **Right:** the mean of the similarities between the gradients of tasks with respect to the shared parameters. Because of the preference we express for fMRI, the sample probabilities for fMRI tasks are much higher. The sampler also learns to favor the more reliable fMRI participants in its sampling. Although the similarities in the sampler are much less granular than the similarities we compute from model parameters, we see that the proto-role tasks are similar to many other tasks, as we also observe in the model-parameter based similarity.

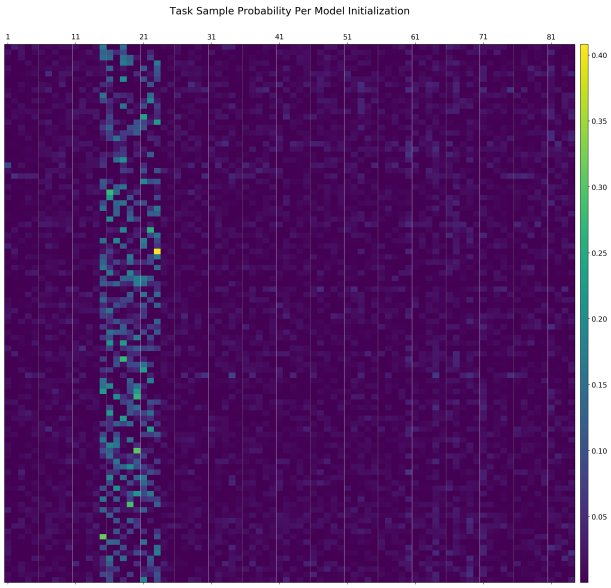
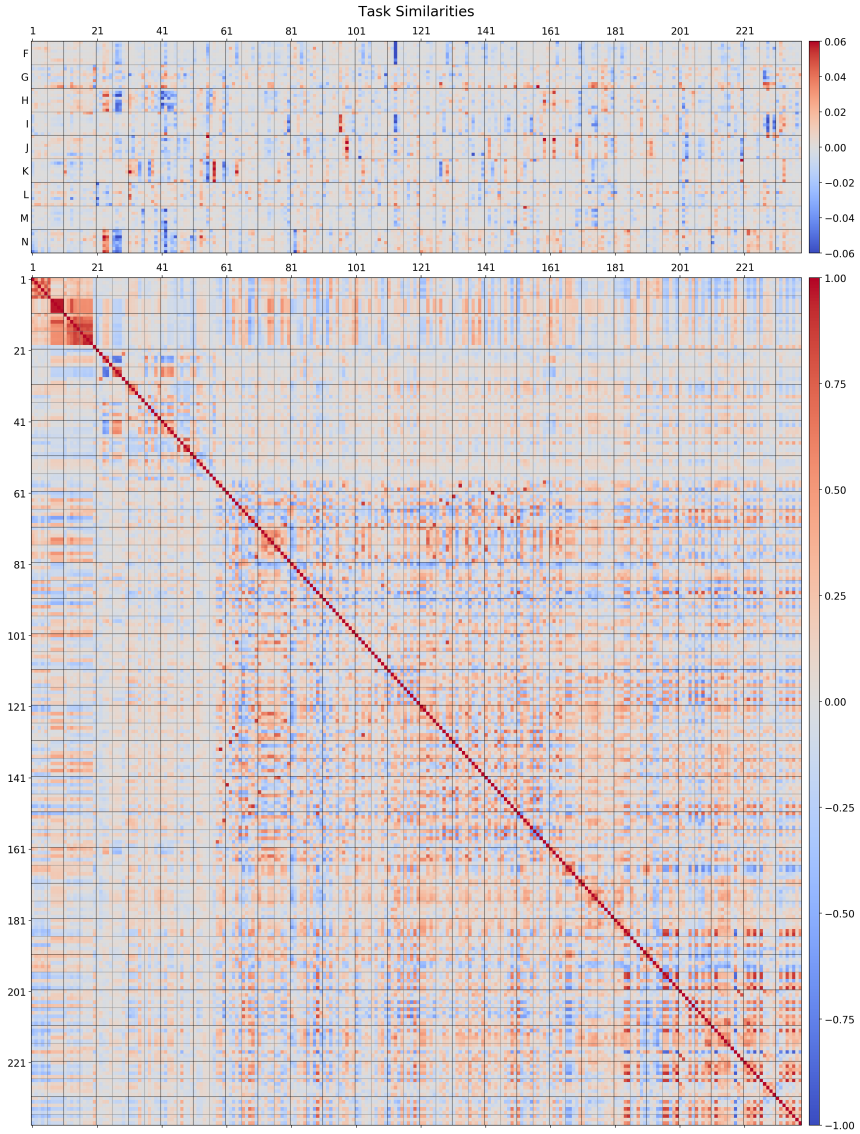
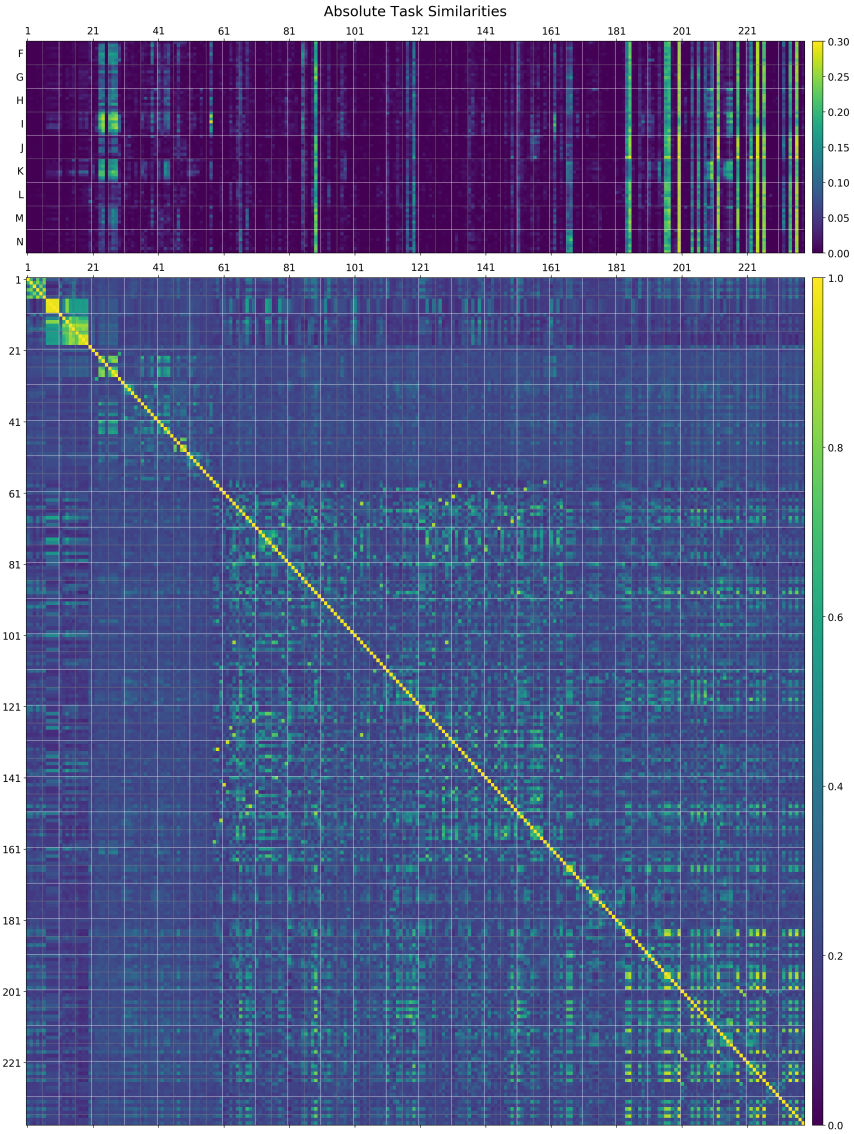


Figure D.13: Probability of a task being sampled for a given model initialization. Each row is one model initialization, and each column shows the probability of a task being sampled at the end of training. The high probability tasks are all fMRI, but for each model initialization, different participants come to dominate the sampling



1. elan
2. lan
3. n400
4. epnp
5. p600
6. pnp
7. ucl.first_fix.
8. ucl.first_pass
9. ucl.right_bnd.
10. ucl.go_past
11. ucl.self_paced.
12. geco.first_fix.
13. geco.fix_cnt.
14. geco.first_pass
15. geco.go_past
16. geco.tot_rd_time.
17. dundee.first_pass
18. dundee.go_past
19. dundee.right_bnd.
20. sentiment
21. bshift
22. obj_num
23. sent_len.0
24. sent_len.1
25. sent_len.2
26. sent_len.3
27. sent_len.4
28. sent_len.5
29. subj_num
30. tc.ADVP_NP_VP_.
31. tc.CC_ADVP_NP_VP_.
32. tc.CC_NP_VP_.
33. tc.IN_NP_VP_.
34. tc.NP_ADVP_VP_.
35. tc.NP_NP_VP_.
36. tc.NP_PP_.
37. tc.NP_VP_.
38. tc.OTHER
39. tc.PP_NP_VP_.
40. tc.RB_NP_VP_.
41. tc.SBAR_NP_VP_.
42. tc.SBAR_VP_.
43. tc.S_CC_S_.
44. tc.S_NP_VP_.
45. tc.S_VP_.
46. tc.VBD_NP_VP_.
47. tc.VP_.
48. tc.WHADVP_SQ_.
49. tc.WHNP_SQ_.
50. tree_depth.10
51. tree_depth.11
52. tree_depth.5
53. tree_depth.6
54. tree_depth.7
55. tree_depth.8
56. tree_depth.9
57. tense
58. .
59. CC
60. CD
61. DT
62. IN
63. JJ
64. MD
65. NN
66. NNP
67. NNS 88,89.
68. PRP 90,91.
69. PRP$ 92,93.
70. RB 94,95.
71. TO 96,97.
72. VB 98,99.
73. VBD 100,101.
74. VBG 102,103.
75. VBN 104,105.
76. VBP 106,107.
77. VBZ 108,109.
78. ADJP 110.
79. ADVP 111.
80. NML 112.
81. NP 113.
82. PP 114.
83. S 115.
84. SBAR 116.
85. TOP
86. VP
87. WHNP
88. sr.ARG0 121,122.
89. sr.ARG1 123,124.
90. sr.ARG2 125,126.
91. sr.ARGM-ADV 127,128.
92. sr.ARGM-DIS 129,130.
93. sr.ARGM-LOC 131,132.
94. sr.ARGM-MNR 133,134.
95. sr.ARGM-MOD 135,136.
96. sr.ARGM-NEG 137,138.
97. sr.ARGM-TMP 139,140.
98. sr.R-ARG0 141,142.
99. ne.CARDINAL 143,144.
100. ne.DATE 145,146.
101. ne.GPE 147,148.
102. ne.LOC 149,150.
103. ne.MONEY 151,152.
104. ne.NORP 153,154.
105. ne.ORDINAL 155,156.
106. ne.ORG 157,158.
107. ne.PERCENT 159,160.
108. ne.PERSON 161,162.
109. ne.TIME 163,164.
110. coref 165,166.
111. dp.advcl 167,168.
112. dp.advmod 169,170.
113. dp.amod 171,172.
114. dp.aux 173,174.
115. dp.case 175,176.
116. dp.cc 177,178.
117. dp.ccomp 179,180.
118. dp.compound 181,182.
119. dp.conj 183,184.
120. dp.cop 185,186.
121. dp.det 187,188.
122. dp.mark 189,190.
123. dp.nmod 191,192.
124. dp.nmod:poss 193,194.
125. dp.nsubj 195,196.
126. dp.nummod 197,198.
127. dp.obj 199,200.
128. dp.obl 201,202.
129. dp.punct 203,204.
130. dp.root 205,206.
131. dp.xcomp 207,208.
132. se.Cause-Eff(e1,e2) 209,210.
133. se.Cause-Eff(e2,e1) 211,212.
134. se.Comp-Whl(e1,e2) 213,214.
135. se.Comp-Whl(e2,e1) 215,216.
136. se.Ctnt-Ctnr(e1,e2) 217,218.
137. se.Ctnt-Ctnr(e2,e1) 219,220.
138. se.Ent-Dest(e1,e2) 221,222.
139. se.Ent-Orig(e1,e2) 223,224.
140. se.Ent-Orig(e2,e1) 225,226.
141. se.Instr-Agn(e1,e2) 227,228.
142. se.Instr-Agn(e2,e1) 229,230.
143. se.Mem-Coll(e2,e1) 231,232.
144. se.Msg-Tpc(e1,e2) 233,234.
145. se.Msg-Tpc(e2,e1) 235,236.
146. se.Other 237,238.
147. se.Pdct-Pdcr(e1,e2)
148. se.Pdct-Pdcr(e2,e1)
149. pr1.awareness
150. pr1.chng_of_st
151. pr1.ex_aft
152. pr1.ex_bef
153. pr1.ex_dur
154. pr1.ex_as_phys
155. pr1.instigation
156. pr1.loc_of_evt
157. pr1.mk_phys_ctct
158. pr1.man_by_another
159. pr1.pred_chng_arg
160. pr1.sentient
161. pr1.volition
162. pr2.awareness
163. pr2.chng_of_st
164. pr2.chng_of_st_cont
165. pr2.instigation
166. pr2.sentient
167. pr2.volition
168. pr2.was_for_benefit

Figure D.14: Task similarities of fMRI summaries to all other task/classes (top), and of all other task/classes to each other (bottom), where classes in multi-class tasks are removed if they appear in less than 1% of examples, and for some tasks only either span 0 or span 1 is shown. The enumerated list shows the order in which the classes are rendered. If a class has two numbers, that indicates span0 and span1 in that order. Horizontal grid lines are shown every 8 items in the top plot to dilineate participants. All other grid lines are spaced 5 classes/tasks to aid in identification. See text for discussion.



1.	elan	22.	obj_num	43.	tc.S_CC_S_	64.	MD	85.	TOP	117.	ne.ORG	155,156.	dp.obj	197,198.	se.Pdct-Pdcr(e2,e1)
2.	lan	23.	sent_len.0	44.	tc.S_NP_VP_	65.	NN	86.	VP	118.	ne.PERCENT	157,158.	dp.obl	199,200.	pr1.awareness
3.	n400	24.	sent_len.1	45.	tc.S_VP_	66.	NNP	87.	WHNP	119.	ne.PERSON	159,160.	dp.punct	201,202.	pr1.chng_of_st
4.	epnp	25.	sent_len.2	46.	tc.VBD_NP_VP_	67.	NNS	88,89.	sr.ARG0	120.	ne.TIME	161,162.	dp.root	203,204.	pr1.ex_aft
5.	p600	26.	sent_len.3	47.	tc.VP_	68.	PRP	90,91.	sr.ARG1	121,122.	coref	163,164.	dp.xcomp	205,206.	pr1.ex_bef
6.	pnp	27.	sent_len.4	48.	tc.WHADVP_SQ_	69.	PRP$	92,93.	sr.ARG2	123,124.	dp.advcl	165,166.	se.Cause-Eff(e1,e2)	207,208.	pr1.ex_dur
7.	ucl.first_fix.	28.	sent_len.5	49.	tc.WHNP_SQ_	70.	RB	94,95.	sr.ARGM-ADV	125,126.	dp.advmod	167,168.	se.Cause-Eff(e2,e1)	209,210.	pr1.ex_as_phys
8.	ucl.first_pass	29.	subj_num	50.	tree_depth.10	71.	TO	96,97.	sr.ARGM-DIS	127,128.	dp.amod	169,170.	se.Comp-Whl(e1,e2)	211,212.	pr1.instigation
9.	ucl.right_bnd.	30.	tc.ADVP_NP_VP_	51.	tree_depth.11	72.	VB	98,99.	sr.ARGM-LOC	129,130.	dp.aux	171,172.	se.Comp-Whl(e2,e1)	213,214.	pr1.loc_of_evt
10.	ucl.go_past	31.	tc.CC_ADVP_NP_VP_	52.	tree_depth.5	73.	VBD	100,101.	sr.ARGM-MNR	131,132.	dp.case	173,174.	se.Ctnt-Ctnr(e1,e2)	215,216.	pr1.mk_phys_ctct
11.	ucl.self_paced.	32.	tc.CC_NP_VP_	53.	tree_depth.6	74.	VBG	102,103.	sr.ARGM-MOD	133,134.	dp.cc	175,176.	se.Ctnt-Ctnr(e2,e1)	217,218.	pr1.man_by_another
12.	geco.first_fix.	33.	tc.IN_NP_VP_	54.	tree_depth.7	75.	VBN	104,105.	sr.ARGM-NEG	135,136.	dp.ccomp	177,178.	se.Ent-Dest(e1,e2)	219,220.	pr1.pred_chng_arg
13.	geco.fix_cnt.	34.	tc.NP_ADVP_VP_	55.	tree_depth.8	76.	VBP	106,107.	sr.ARGM-TMP	137,138.	dp.compound	179,180.	se.Ent-Orig(e1,e2)	221,222.	pr1.sentient
14.	geco.first_pass	35.	tc.NP_NP_VP_	56.	tree_depth.9	77.	VBZ	108,109.	sr.R-ARG0	139,140.	dp.conj	181,182.	se.Ent-Orig(e2,e1)	223,224.	pr1.volition
15.	geco.go_past	36.	tc.NP_PP_	57.	tense	78.	ADJP	110.	ne.CARDINAL	141,142.	dp.cop	183,184.	se.Instr-Agn(e1,e2)	225,226.	pr2.awareness
16.	geco.tot_rd_time.	37.	tc.NP_VP_	58.	.	79.	ADVP	111.	ne.DATE	143,144.	dp.det	185,186.	se.Instr-Agn(e2,e1)	227,228.	pr2.chng_of_st
17.	dundee.first_pass	38.	tc.OTHER	59.	CC	80.	NML	112.	ne.GPE	145,146.	dp.mark	187,188.	se.Mem-Coll(e2,e1)	229,230.	pr2.chng_of_st_cont
18.	dundee.go_past	39.	tc.PP_NP_VP_	60.	CD	81.	NP	113.	ne.LOC	147,148.	dp.nmod	189,190.	se.Msg-Tpc(e1,e2)	231,232.	pr2.instigation
19.	dundee.right_bnd.	40.	tc.RB_NP_VP_	61.	DT	82.	PP	114.	ne.MONEY	149,150.	dp.nmod:poss	191,192.	se.Msg-Tpc(e2,e1)	233,234.	pr2.sentient
20.	sentiment	41.	tc.SBAR_NP_VP_	62.	IN	83.	S	115.	ne.NORP	151,152.	dp.nsubj	193,194.	se.Other	235,236.	pr2.volition
21.	bshift	42.	tc.SBAR_VP_	63.	JJ	84.	SBAR	116.	ne.ORDINAL	153,154.	dp.nummod	195,196.	se.Pdct-Pdcr(e1,e2)	237,238.	pr2.was_for_benefit

Figure D.15: Absolute task similarities of fMRI summaries to all other task/classes (top), and of all other task/classes to each other (bottom), where classes in multi-class tasks are removed if they appear in less than 1% of examples, and for some tasks only either span 0 or span 1 is shown. The enumerated list shows the order in which the classes are rendered. If a class has two numbers, that indicates span0 and span1 in that order. Horizontal grid lines are shown every 8 items in the top plot to dilineate participants. All other grid lines are spaced 5 classes/tasks to aid in identification. See text for discussion.

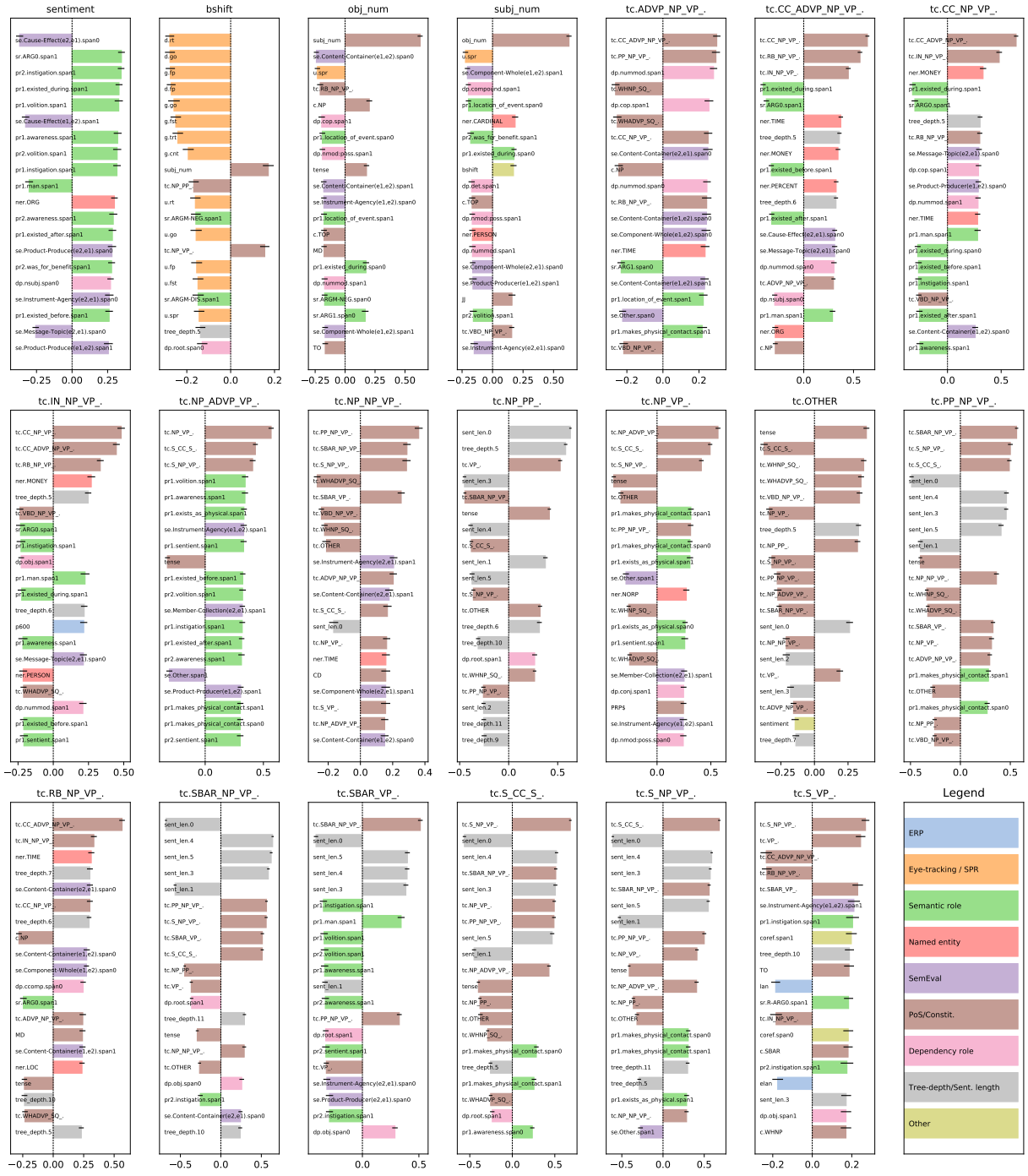


Figure D.16: Similarities between NLP tasks and other tasks (part 1).

Figure D.17: Similarities between NLP tasks and other tasks (part 2).

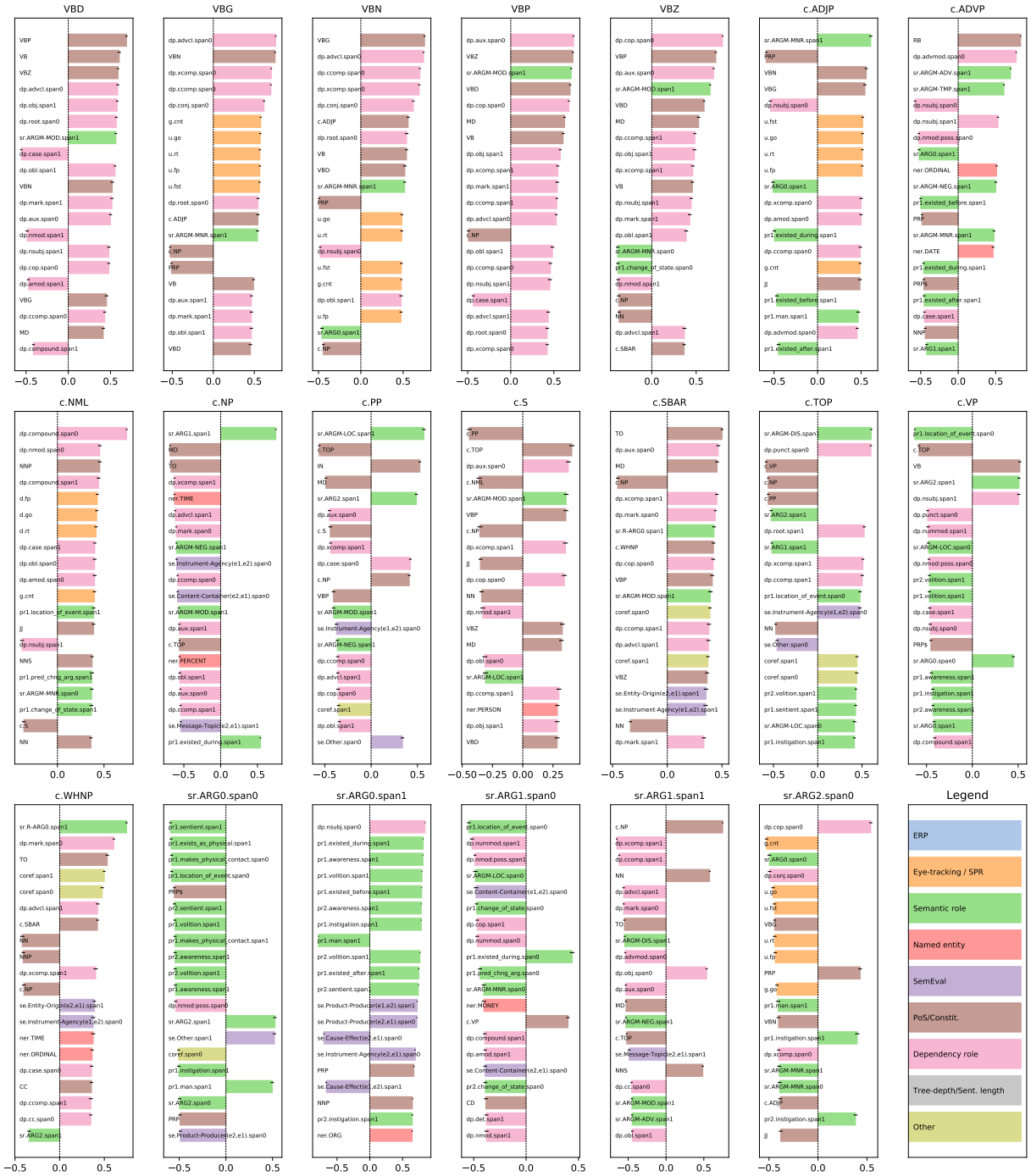


Figure D.18: Similarities between NLP tasks and other tasks (part 3).

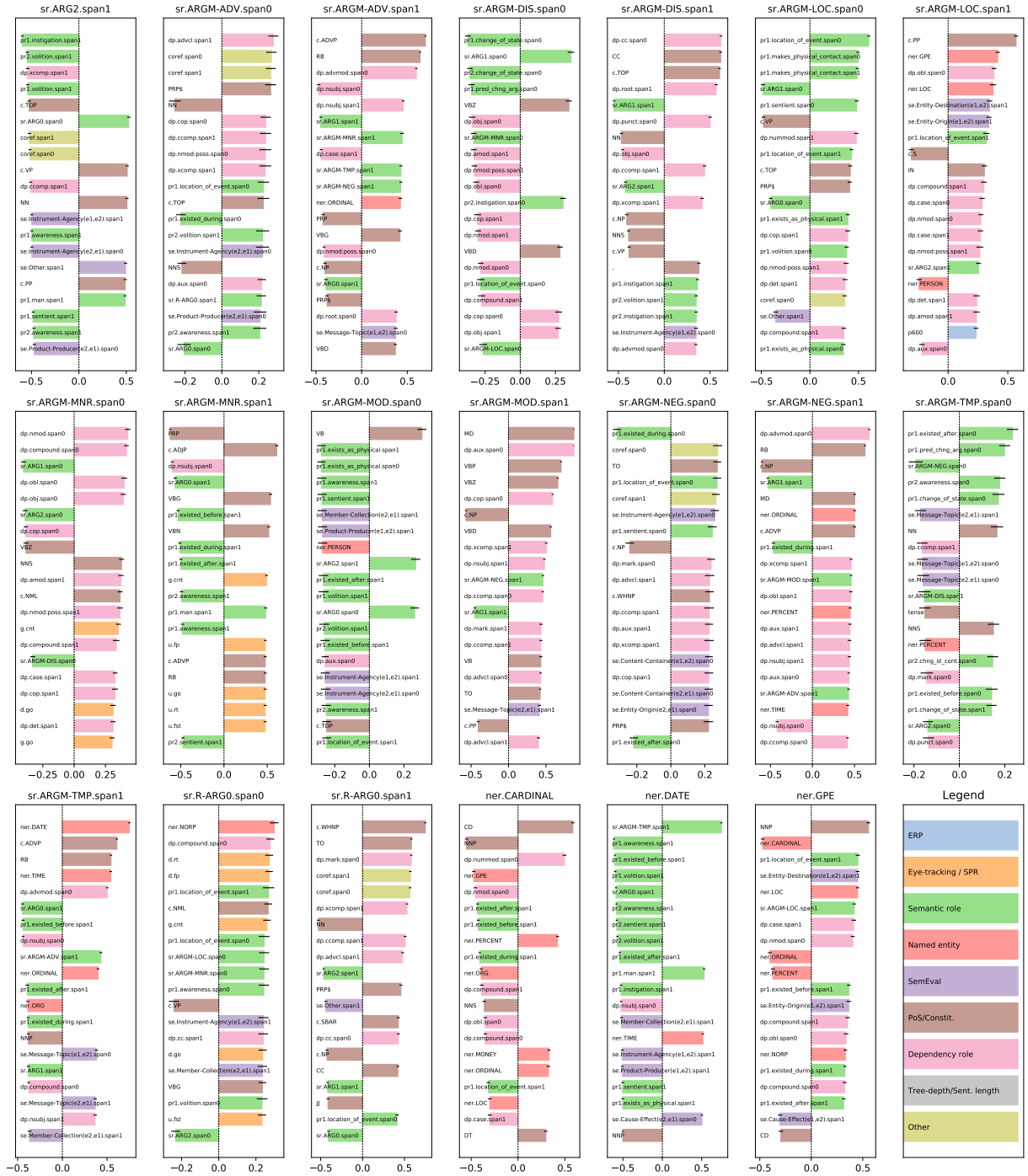


Figure D.19: Similarities between NLP tasks and other tasks (part 4).

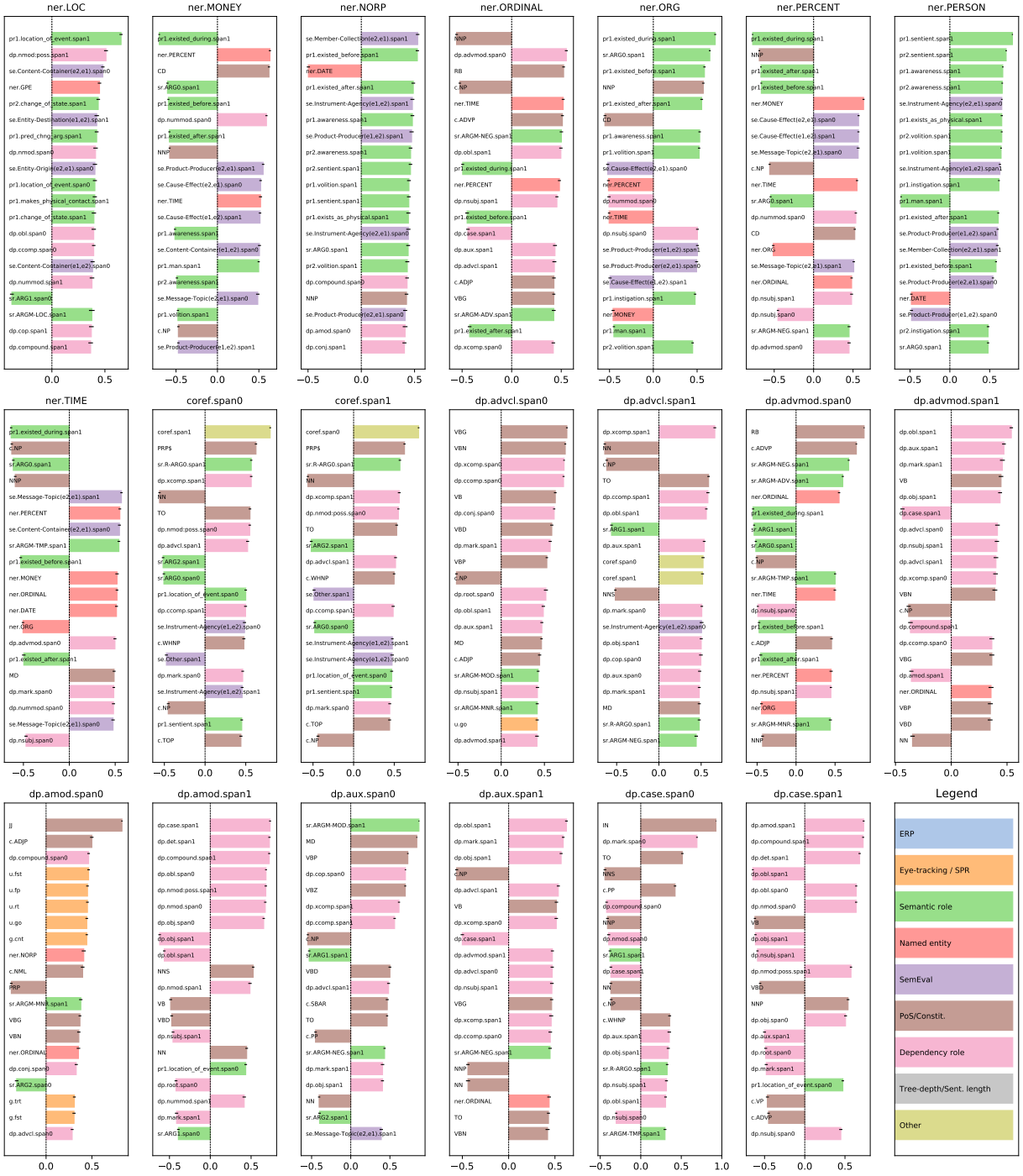


Figure D.20: Similarities between NLP tasks and other tasks (part 5).

Figure D.21: Similarities between NLP tasks and other tasks (part 6).

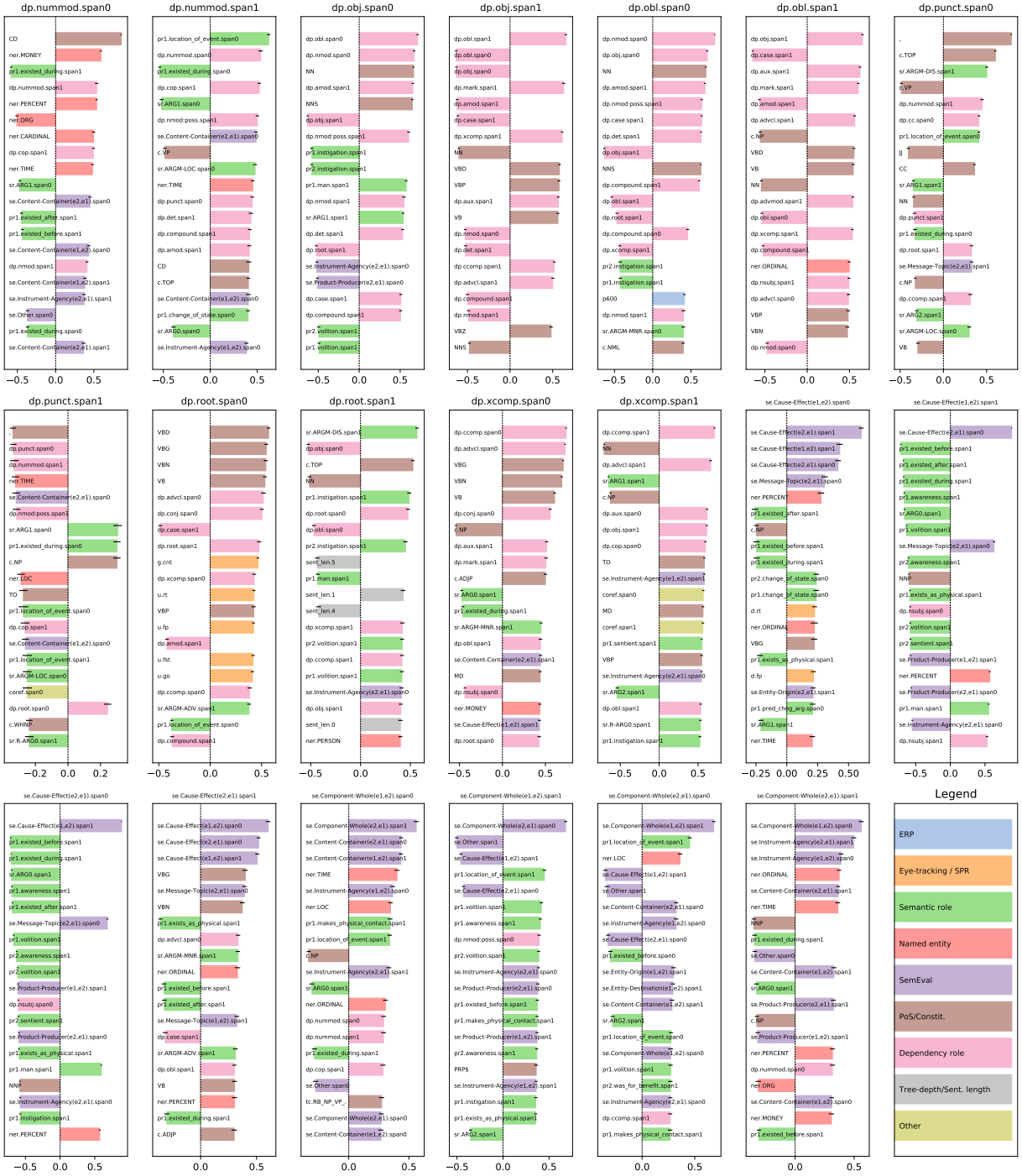


Figure D.22: Similarities between NLP tasks and other tasks (part 7).

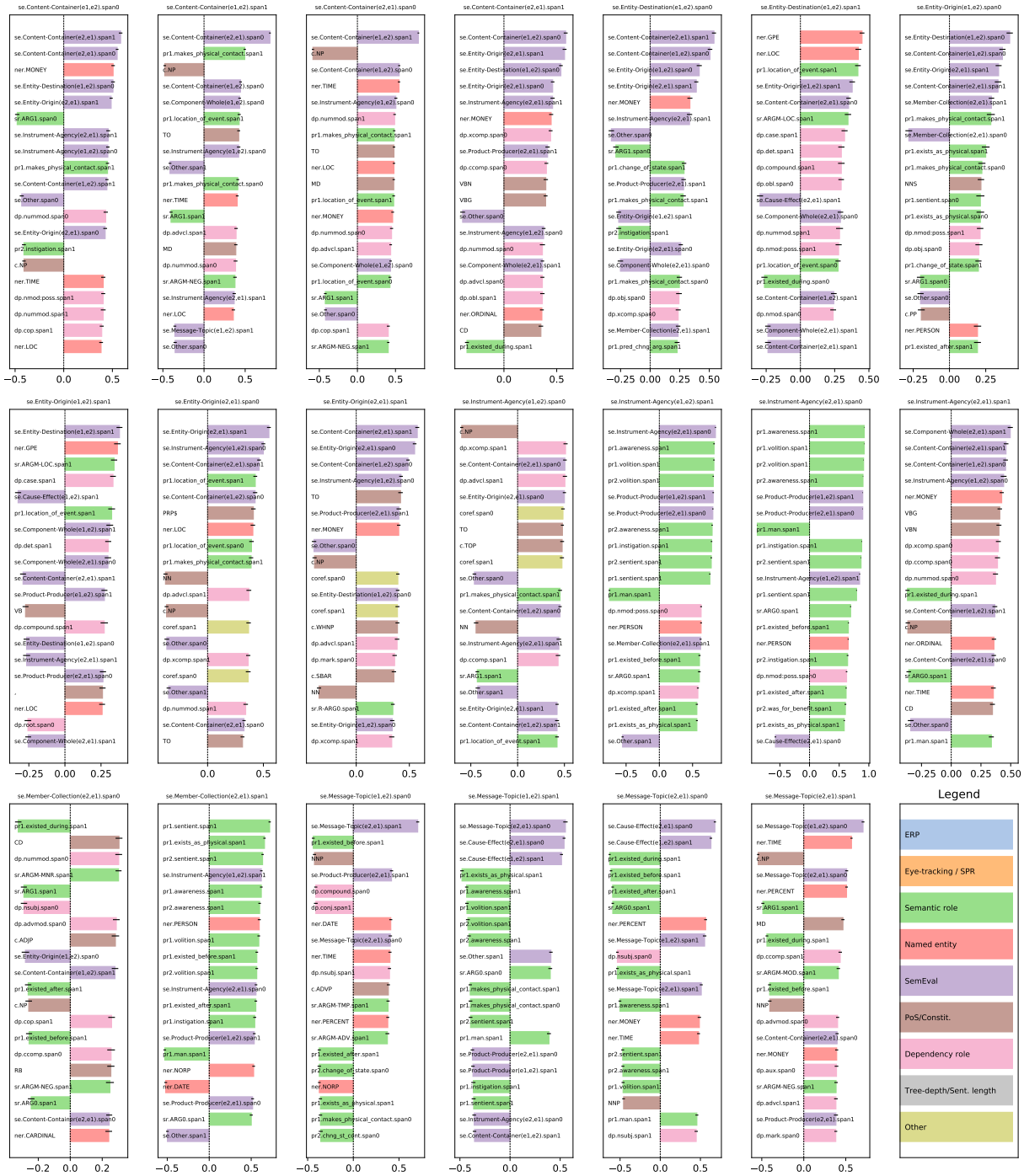


Figure D.23: Similarities between NLP tasks and other tasks (part 8).

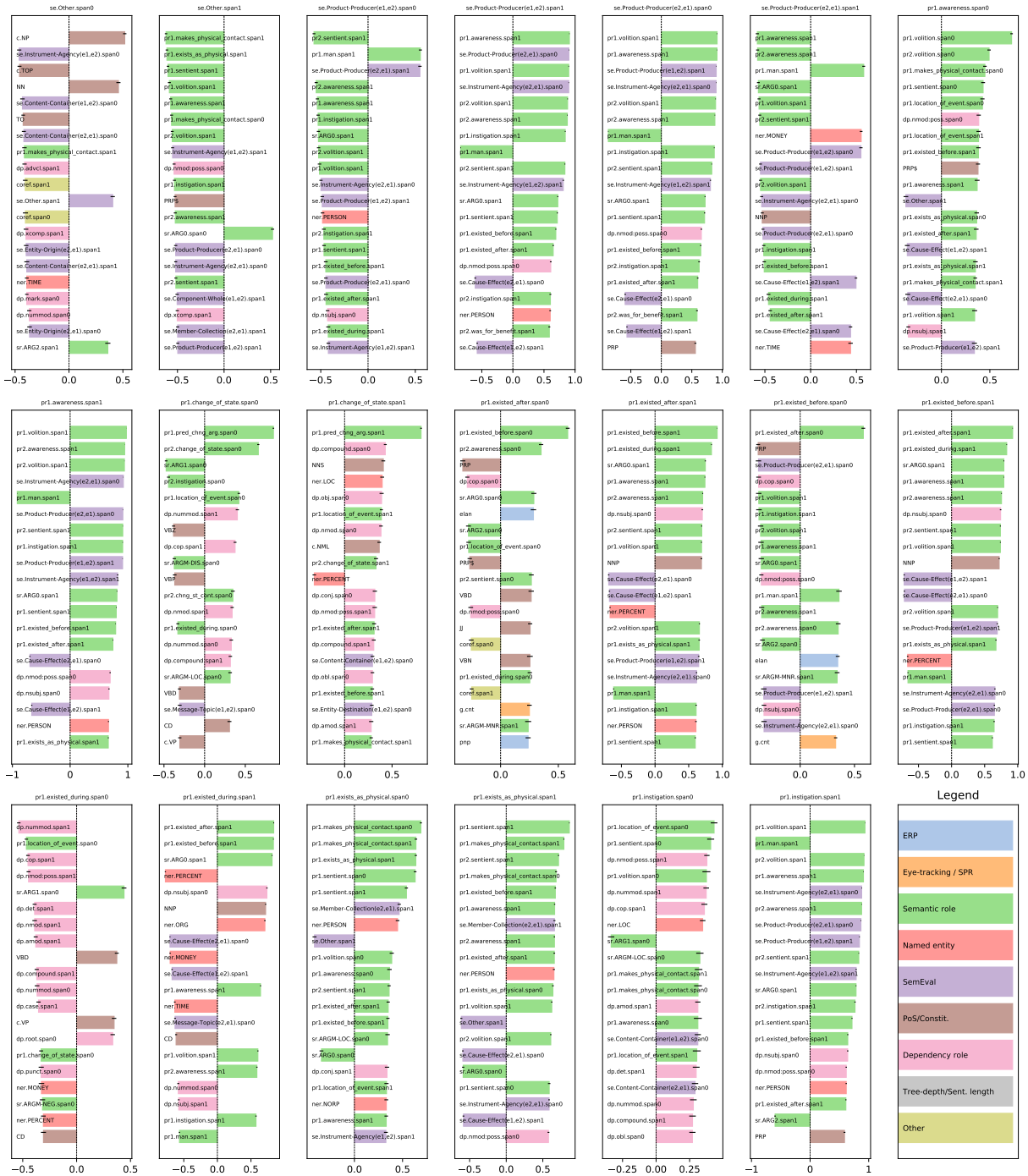


Figure D.24: Similarities between NLP tasks and other tasks (part 9).

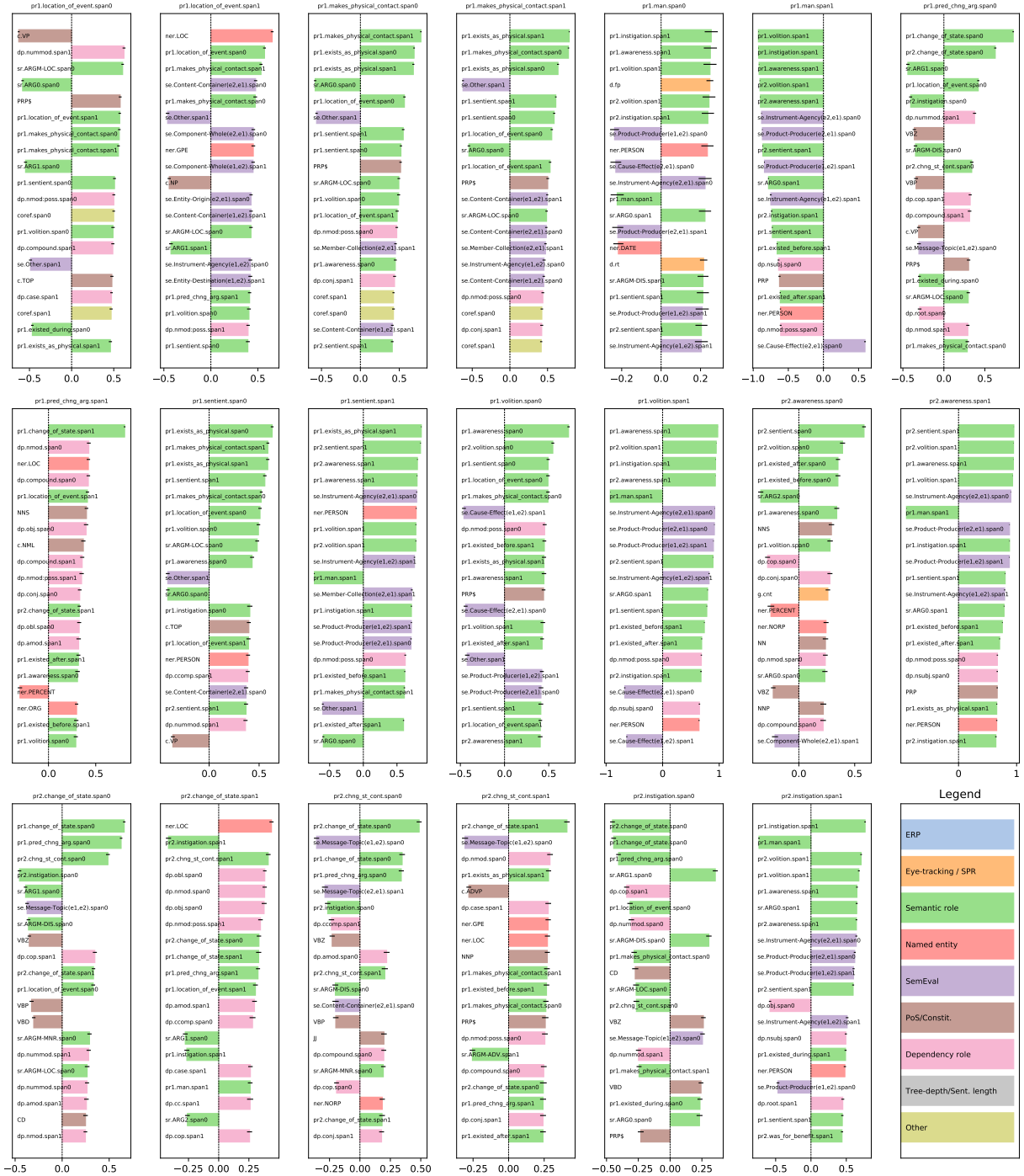


Figure D.25: Similarities between NLP tasks and other tasks (part 10).

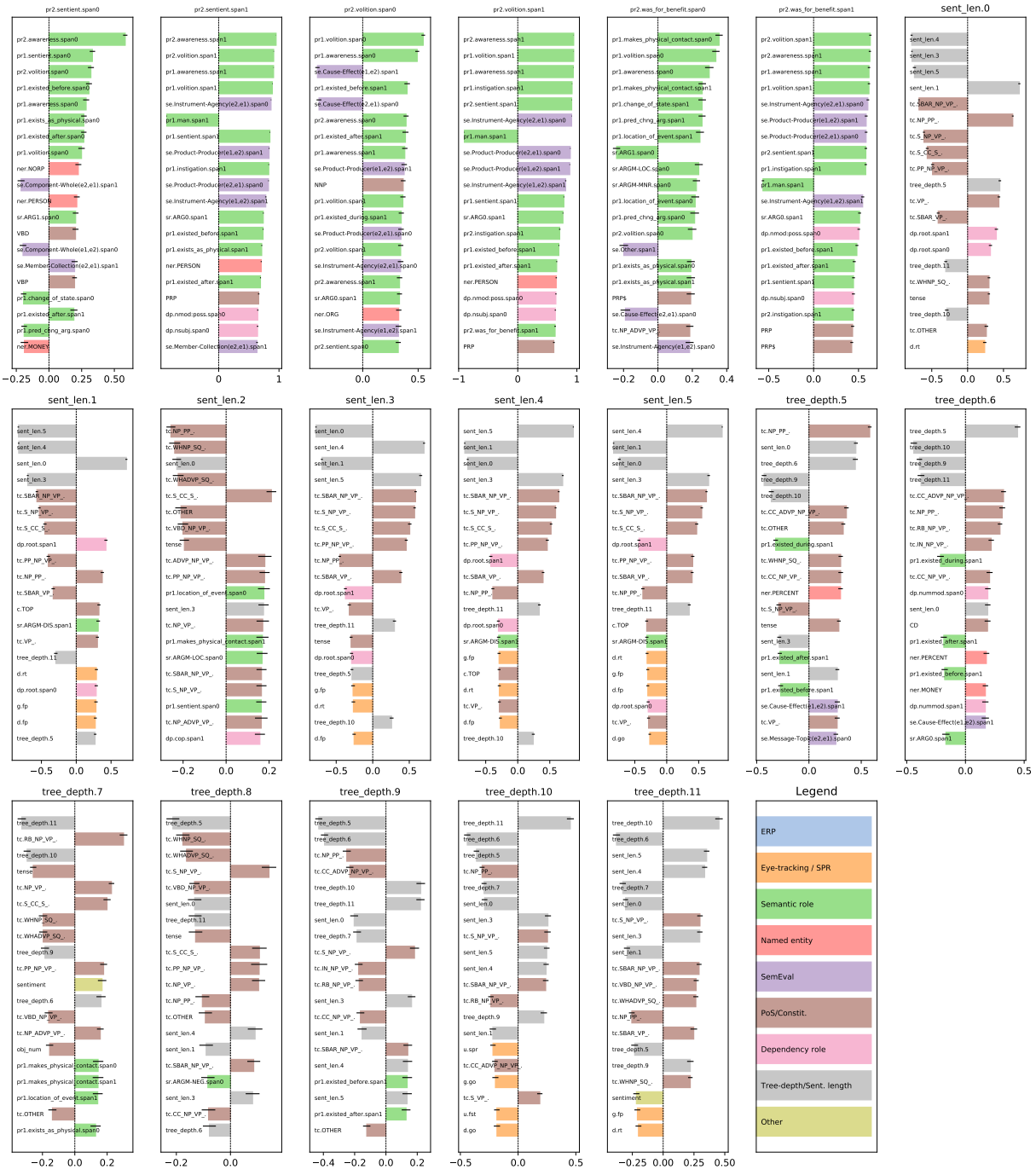


Figure D.26: Similarities between NLP tasks and other tasks (part 11).

Figure D.27: 20 largest absolute task similarities for the summary of each region of interest for participant G as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

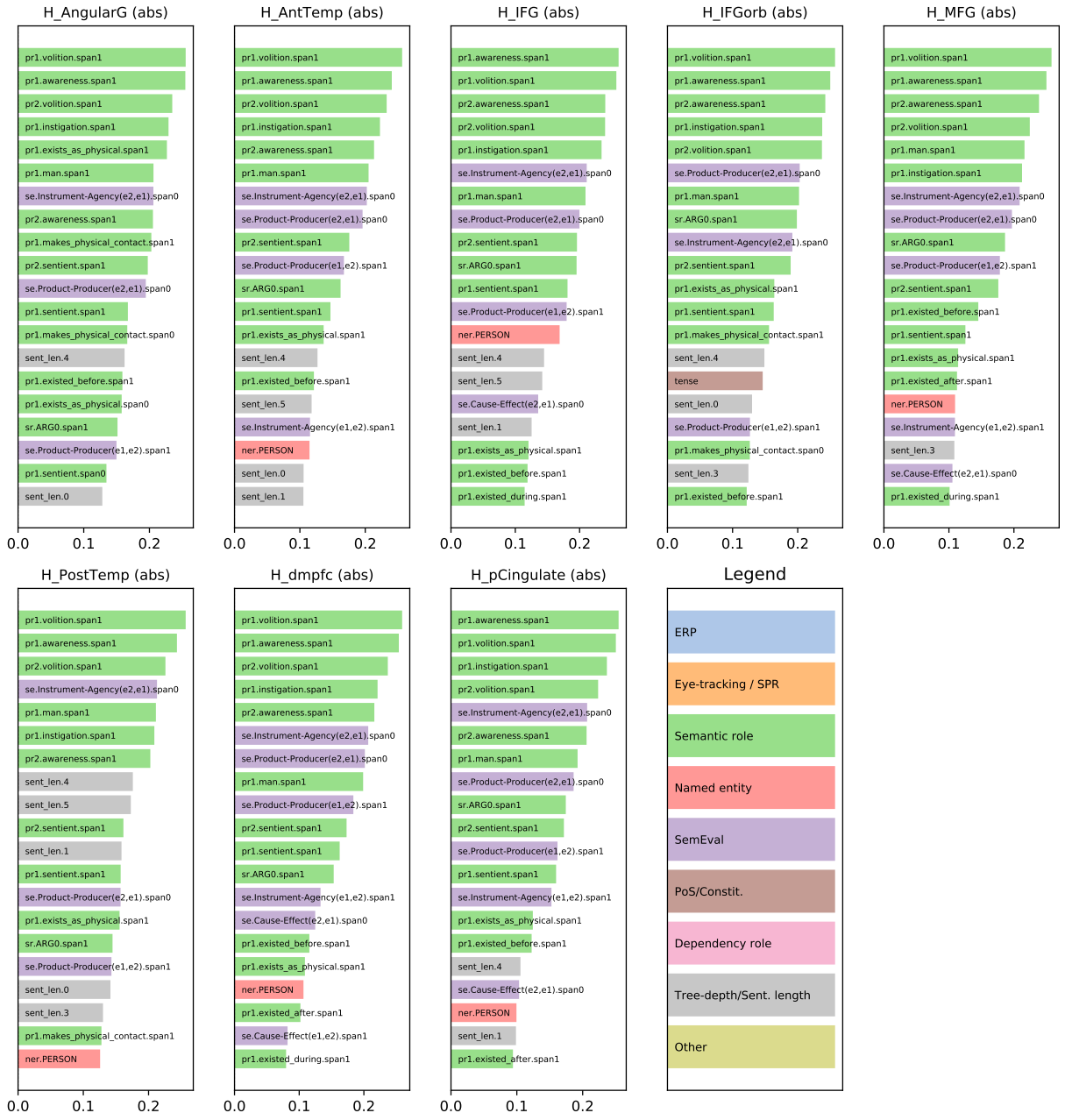


Figure D.28: 20 largest absolute task similarities for the summary of each region of interest for participant H as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

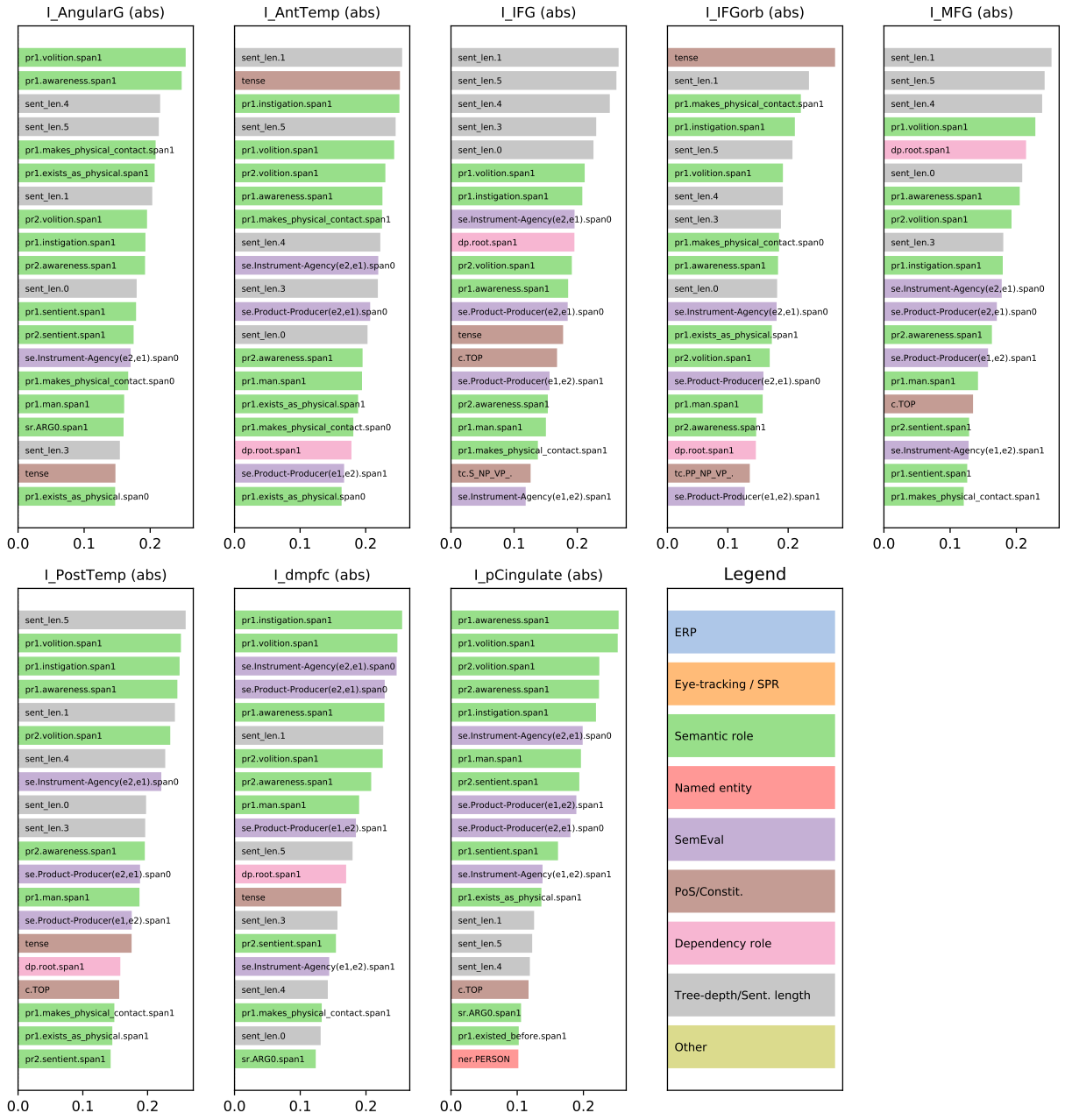


Figure D.29: 20 largest absolute task similarities for the summary of each region of interest for participant I as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

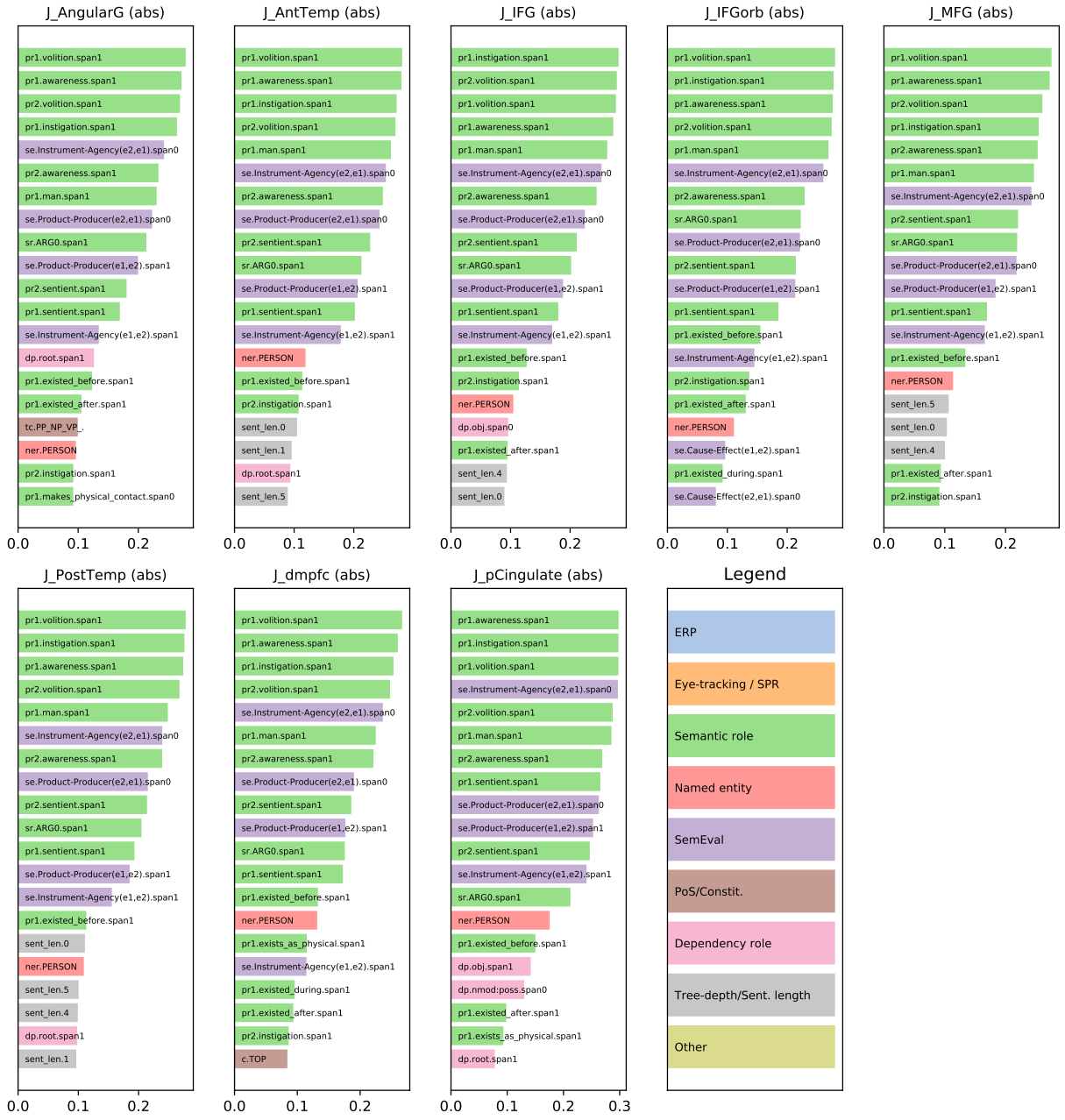


Figure D.30: 20 largest absolute task similarities for the summary of each region of interest for participant J as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

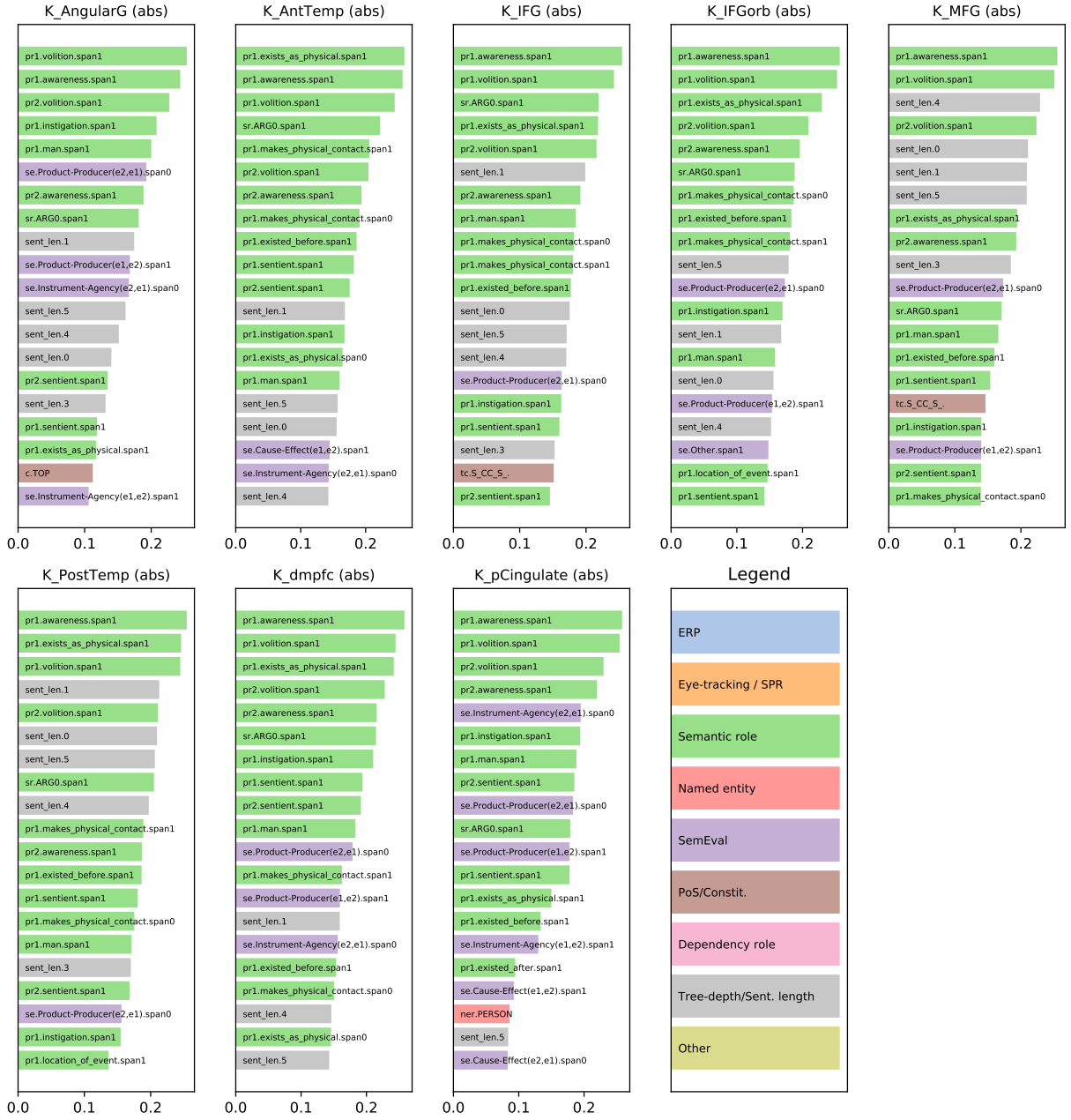


Figure D.31: 20 largest absolute task similarities for the summary of each region of interest for participant K as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

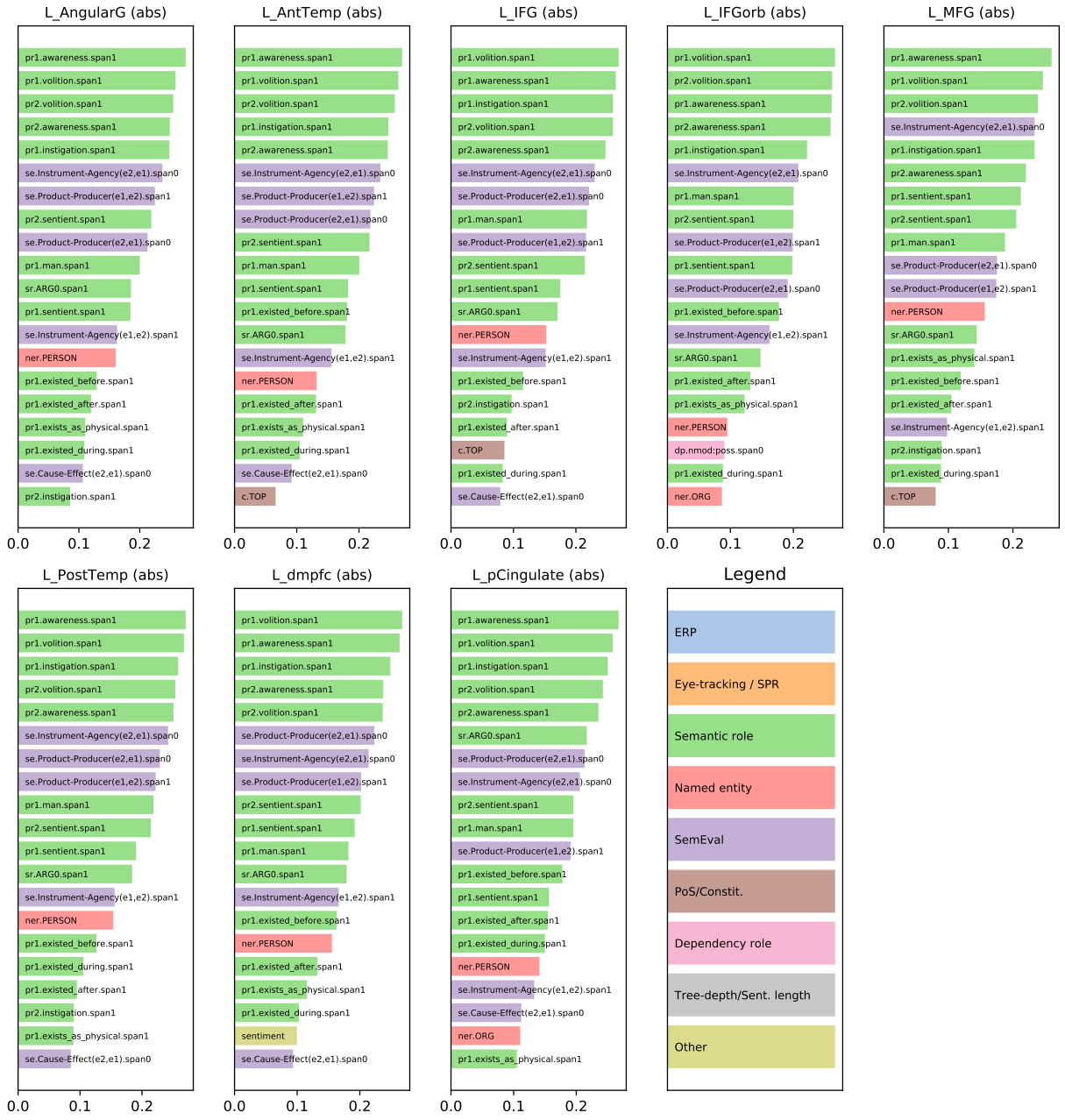


Figure D.32: 20 largest absolute task similarities for the summary of each region of interest for participant L as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

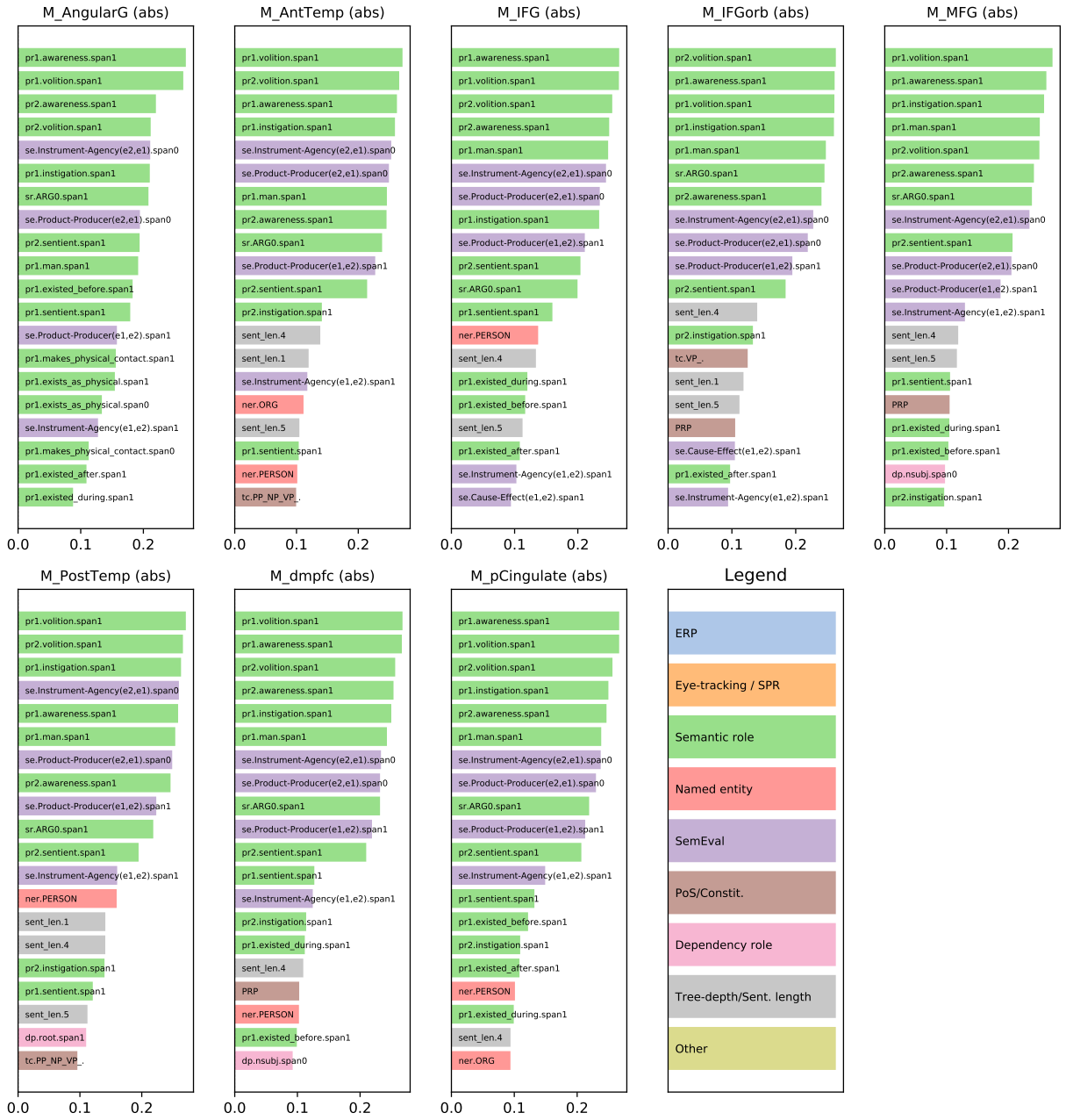


Figure D.33: 20 largest absolute task similarities for the summary of each region of interest for participant M as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

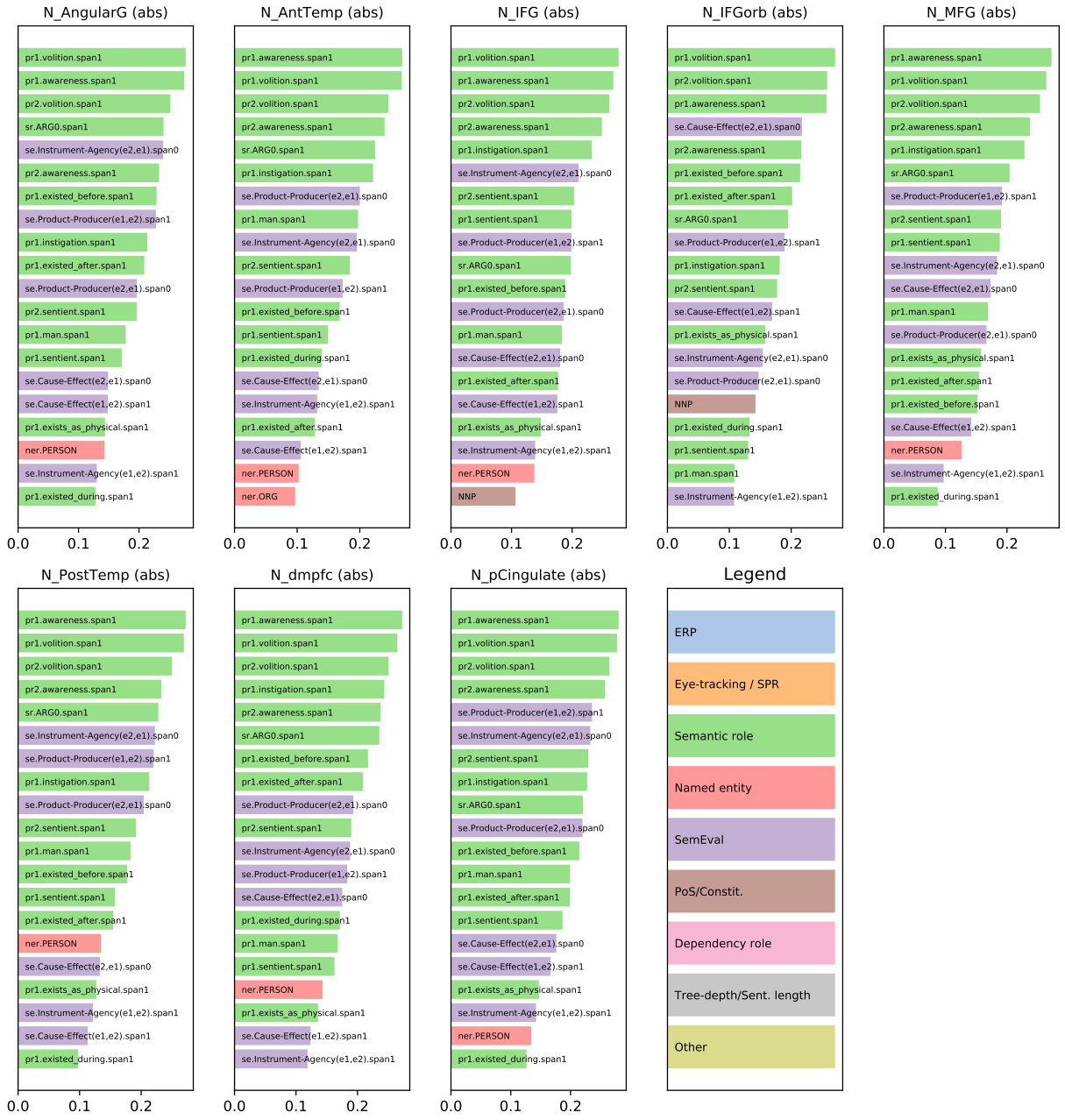


Figure D.34: 20 largest absolute task similarities for the summary of each region of interest for participant N as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

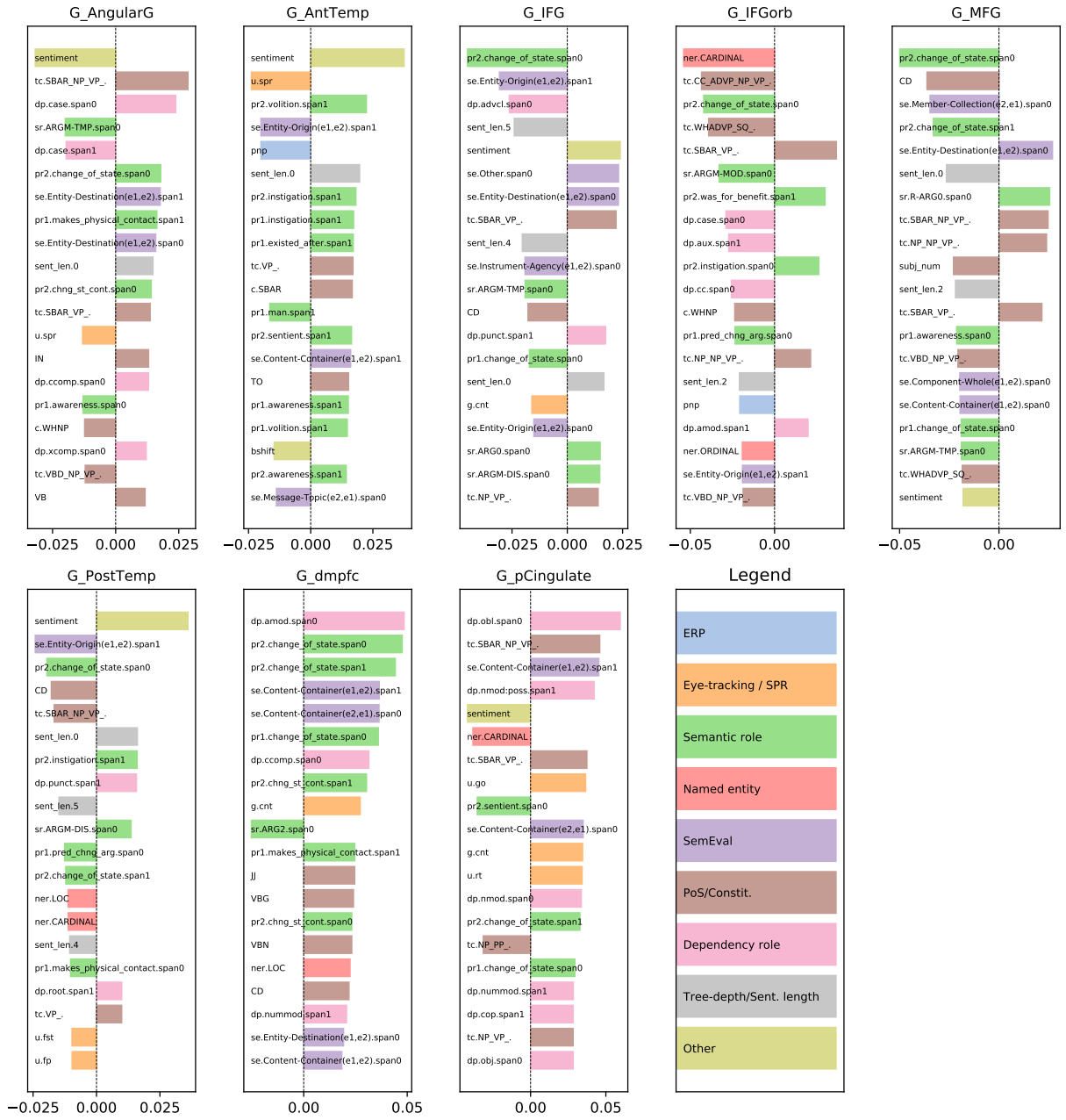


Figure D.35: 20 largest magnitude task similarities for the summary of each region of interest for participant G as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

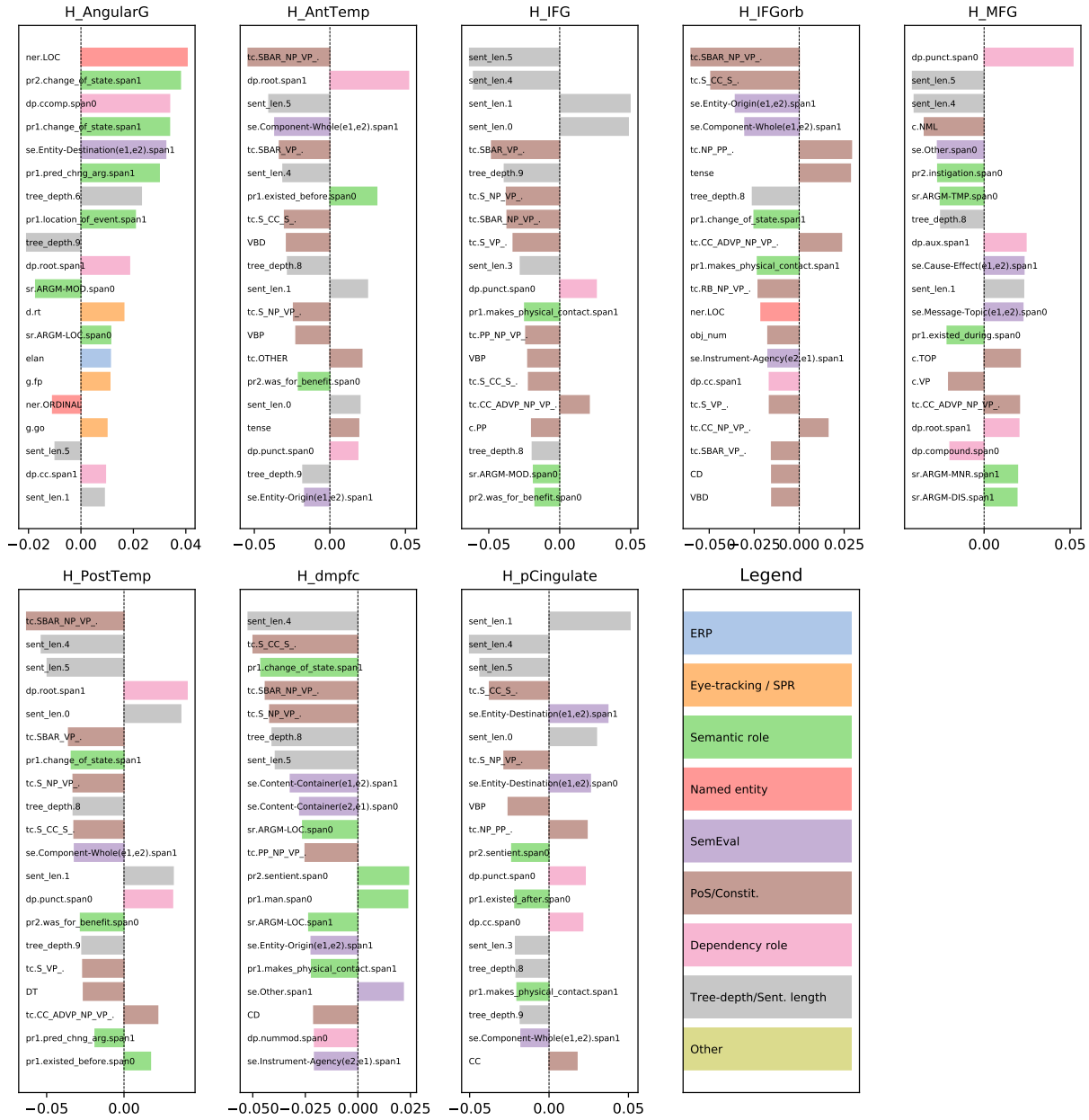


Figure D.36: 20 largest magnitude task similarities for the summary of each region of interest for participant H as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Figure D.37: 20 largest magnitude task similarities for the summary of each region of interest for participant I as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Figure D.38: 20 largest magnitude task similarities for the summary of each region of interest for participant J as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Figure D.39: 20 largest magnitude task similarities for the summary of each region of interest for participant K as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Figure D.40: 20 largest magnitude task similarities for the summary of each region of interest for participant L as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

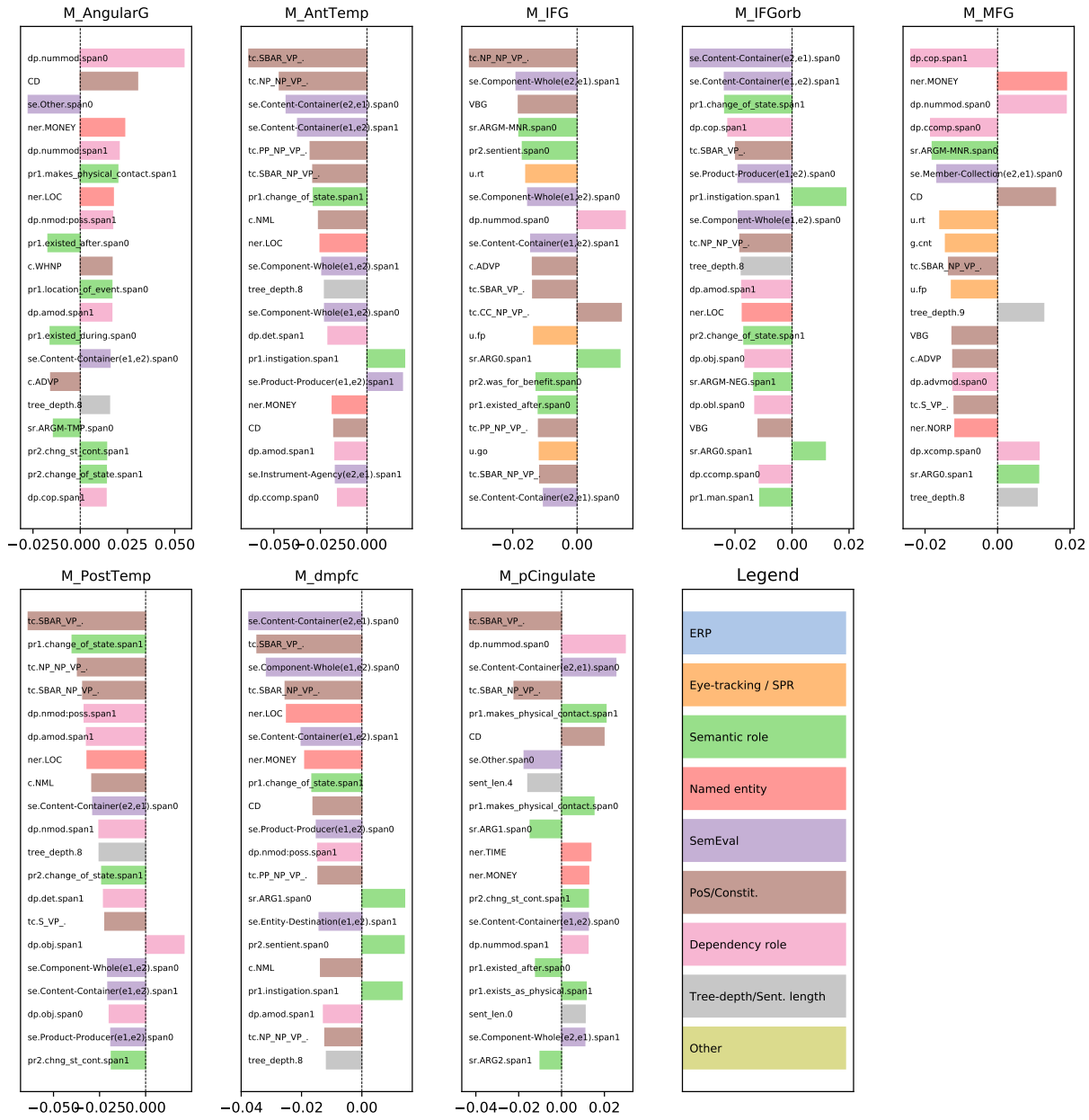


Figure D.41: 20 largest magnitude task similarities for the summary of each region of interest for participant M as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Figure D.42: 20 largest magnitude task similarities for the summary of each region of interest for participant N as computed in section 5.3.3. Bar colors indicate the broad category of each task that fMRI is similar to.

Appendix E

fMRI Voxel-Level Similarities

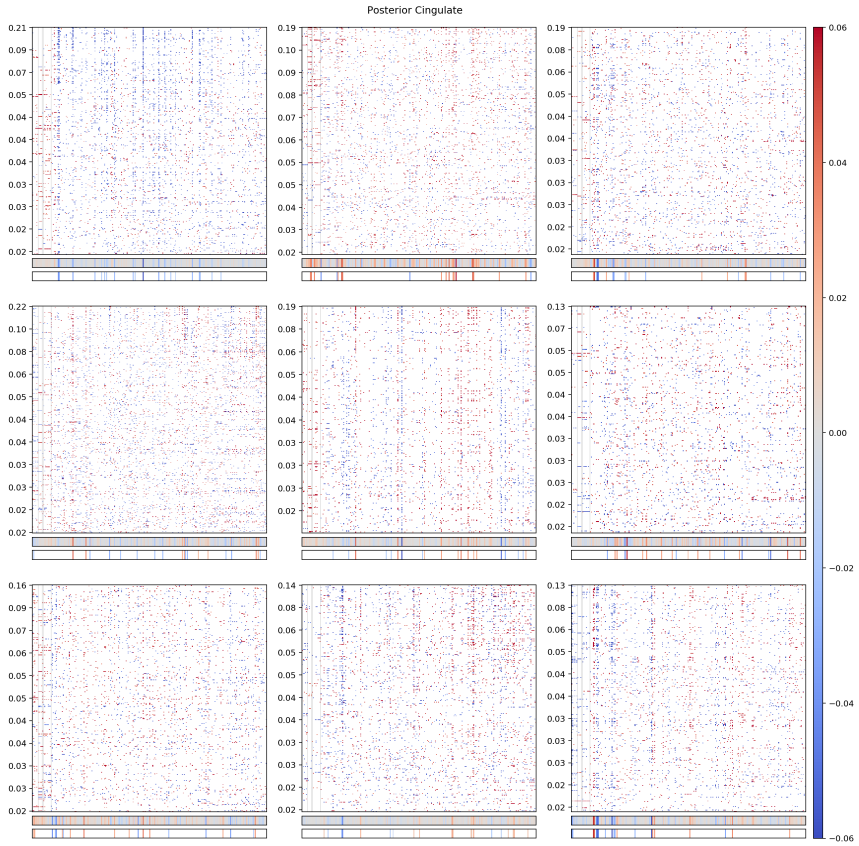


Figure E.1: Task similarities of each voxel within the posterior cingulate as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

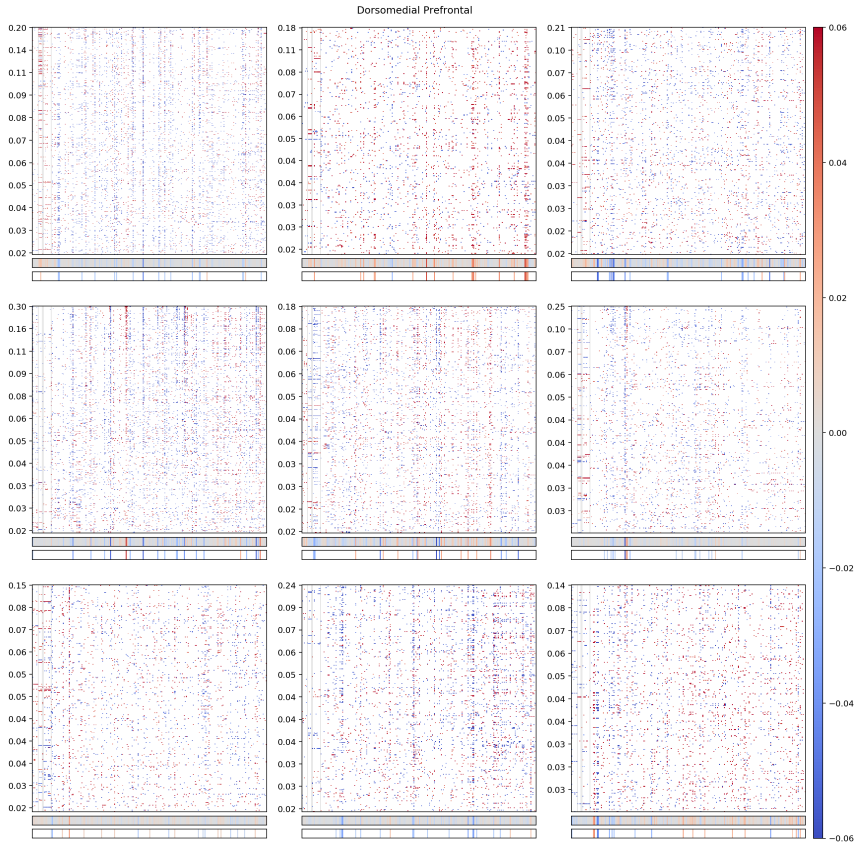


Figure E.2: Task similarities of each voxel within the dorsomedial prefrontal cortex as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

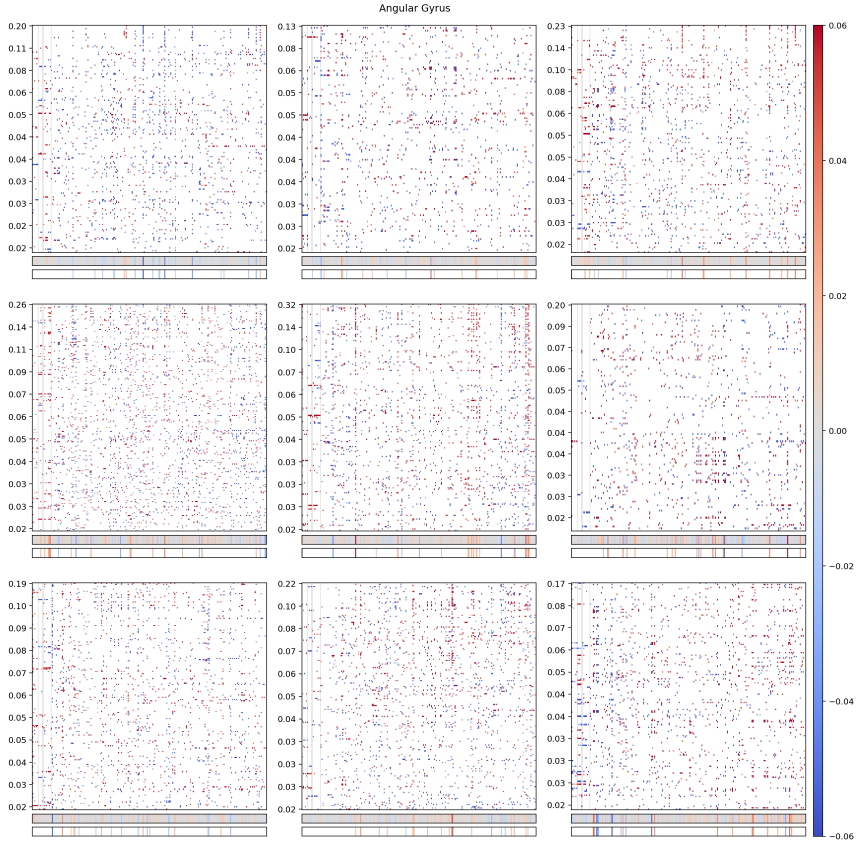


Figure E.3: Task similarities of each voxel within the angular gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

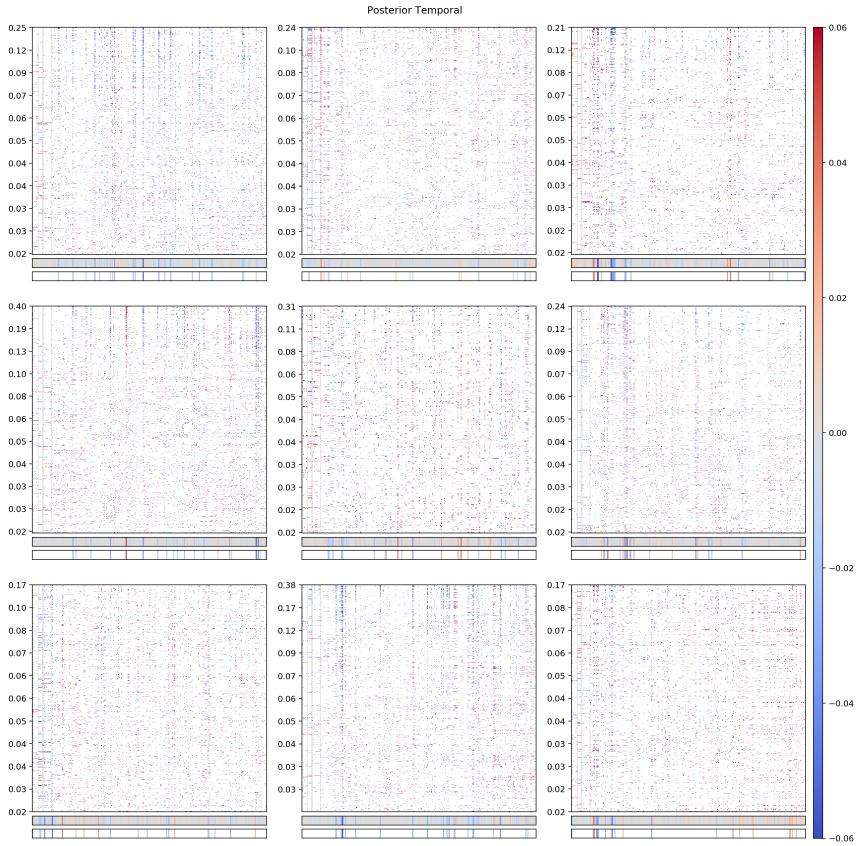


Figure E.4: Task similarities of each voxel within the posterior temporal lobe as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

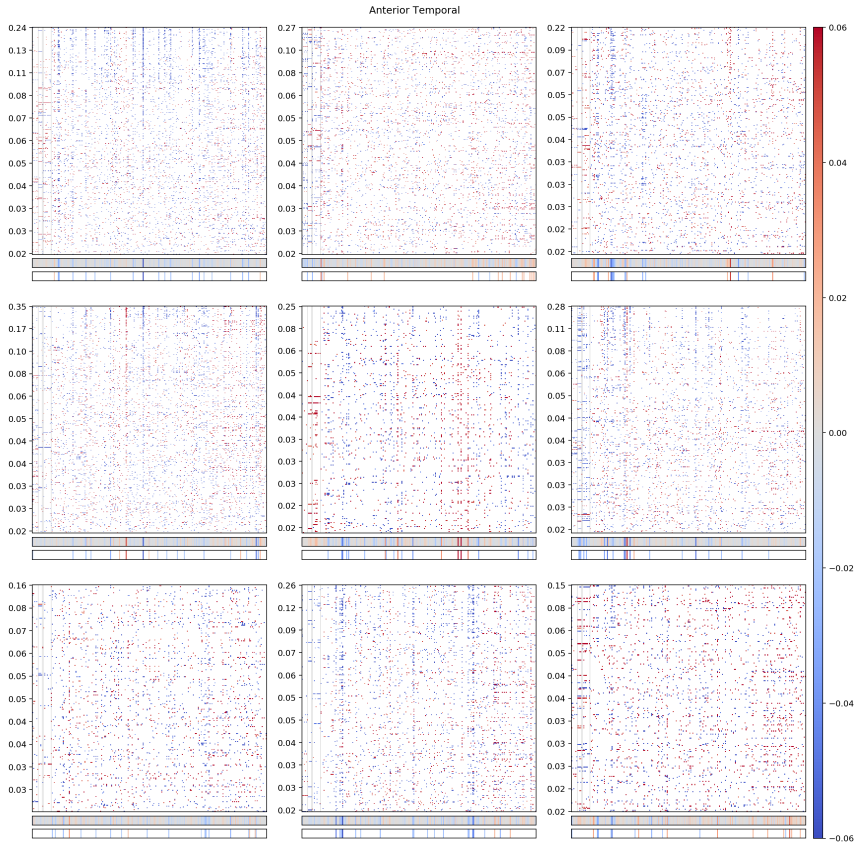


Figure E.5: Task similarities of each voxel within the anterior temporal lobe as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

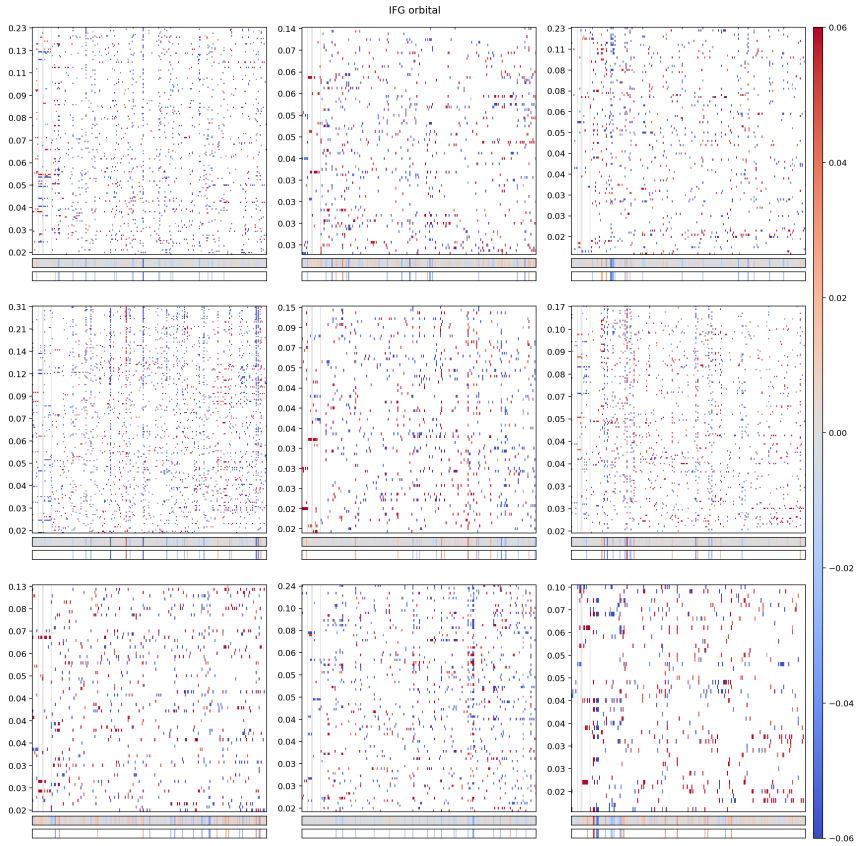


Figure E.6: Task similarities of each voxel within the orbital portion of inferior frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

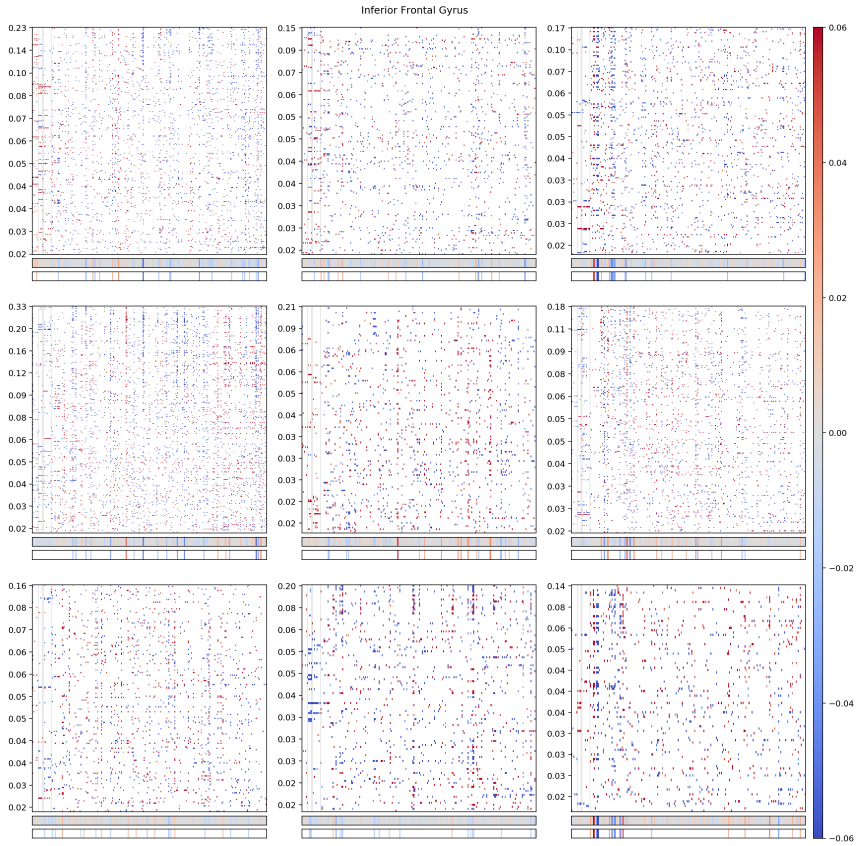


Figure E.7: Task similarities of each voxel within the non-orbital part of inferior frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

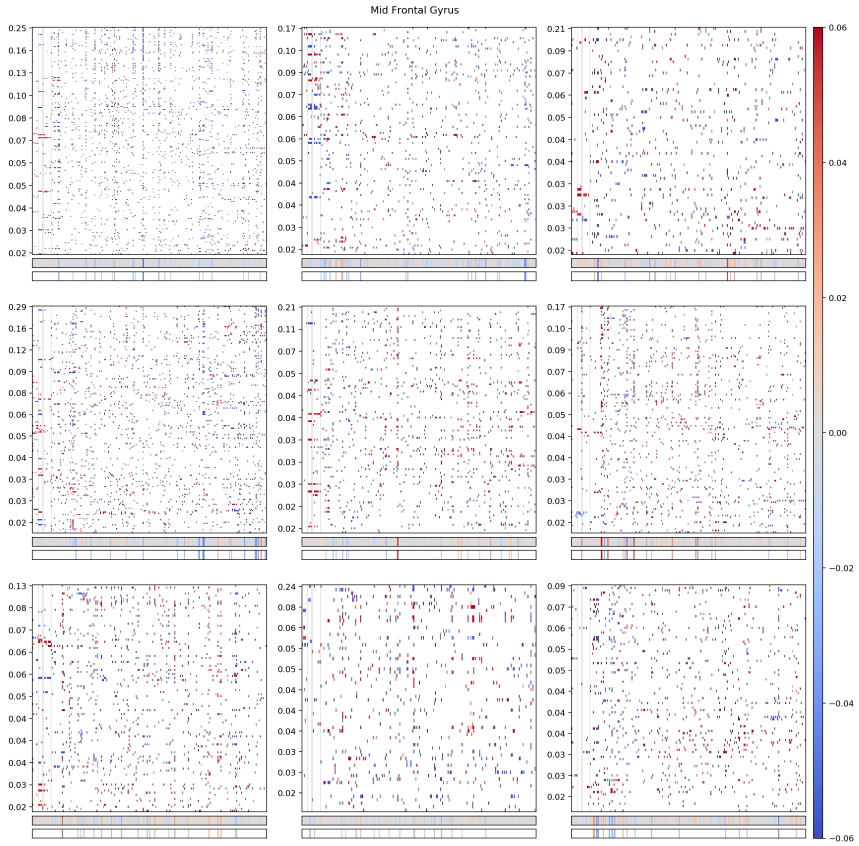


Figure E.8: Task similarities of each voxel within the mid frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

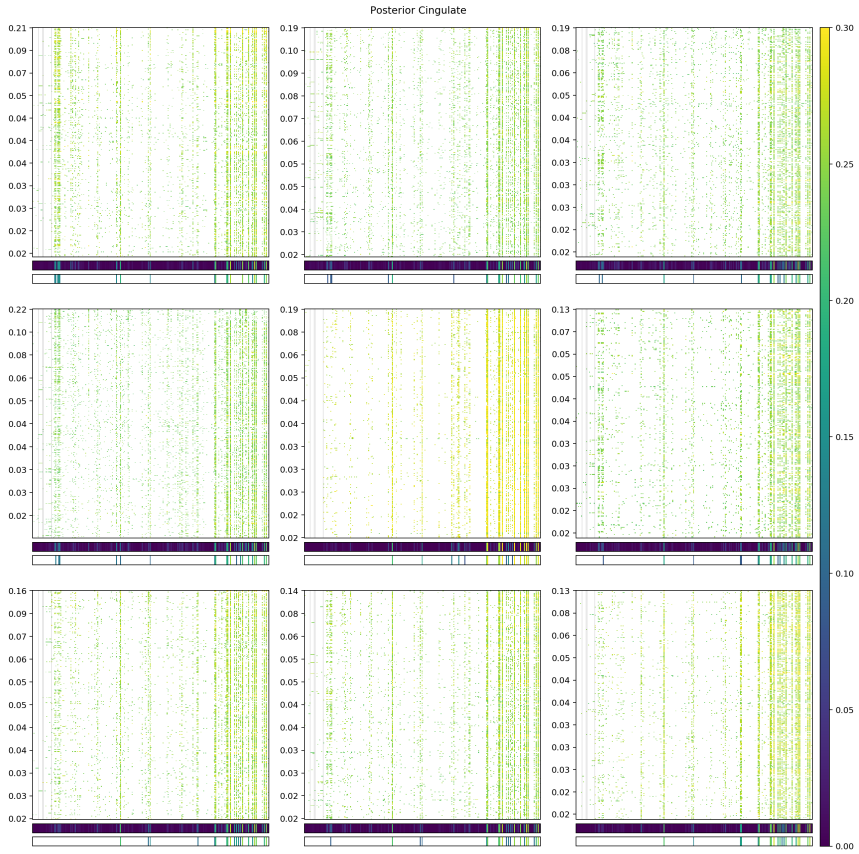


Figure E.9: Absolute task similarities of each voxel within the posterior cingulate as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

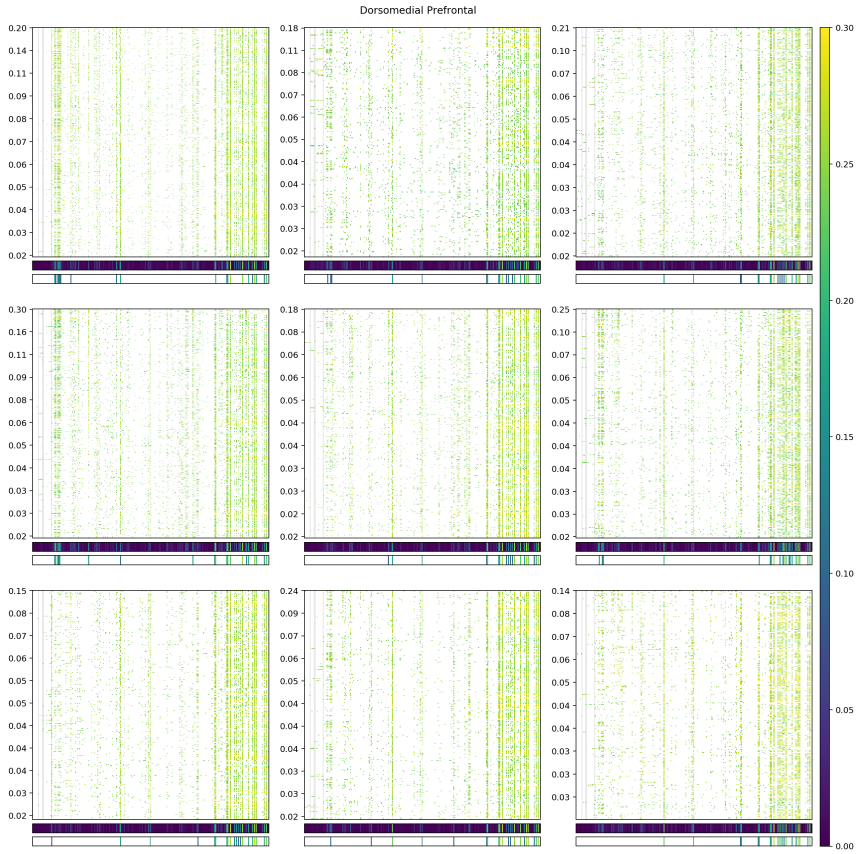


Figure E.10: Absolute task similarities of each voxel within the dorsomedial prefrontal cortex as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

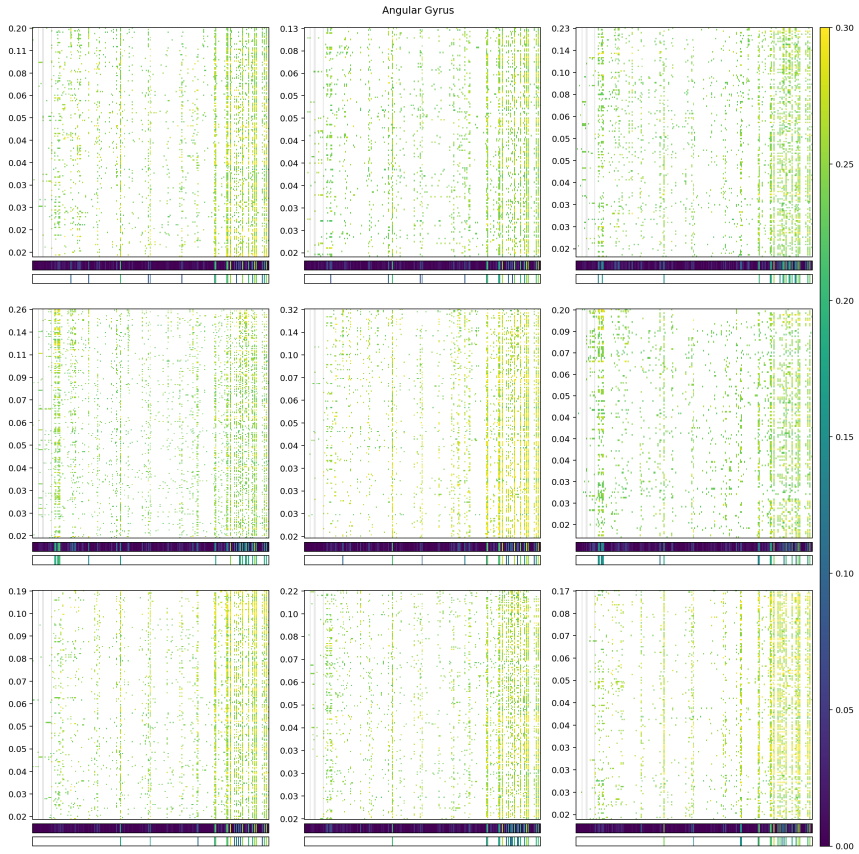


Figure E.11: Absolute task similarities of each voxel within the angular gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

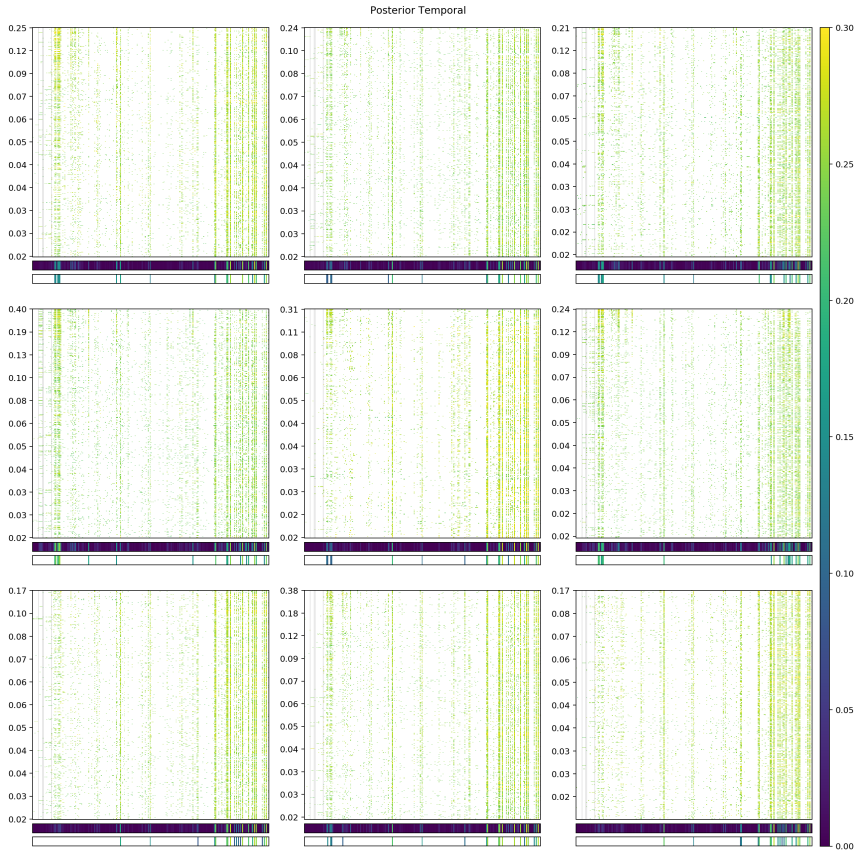


Figure E.12: Absolute task similarities of each voxel within the posterior temporal lobe as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

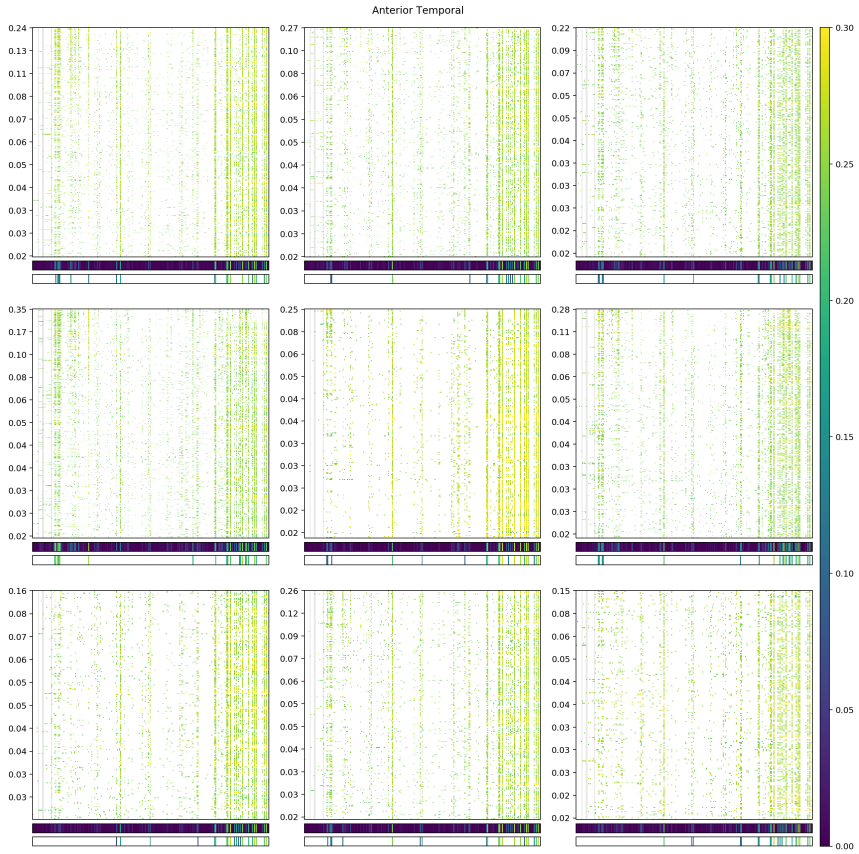


Figure E.13: Absolute task similarities of each voxel within the anterior temporal lobe as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

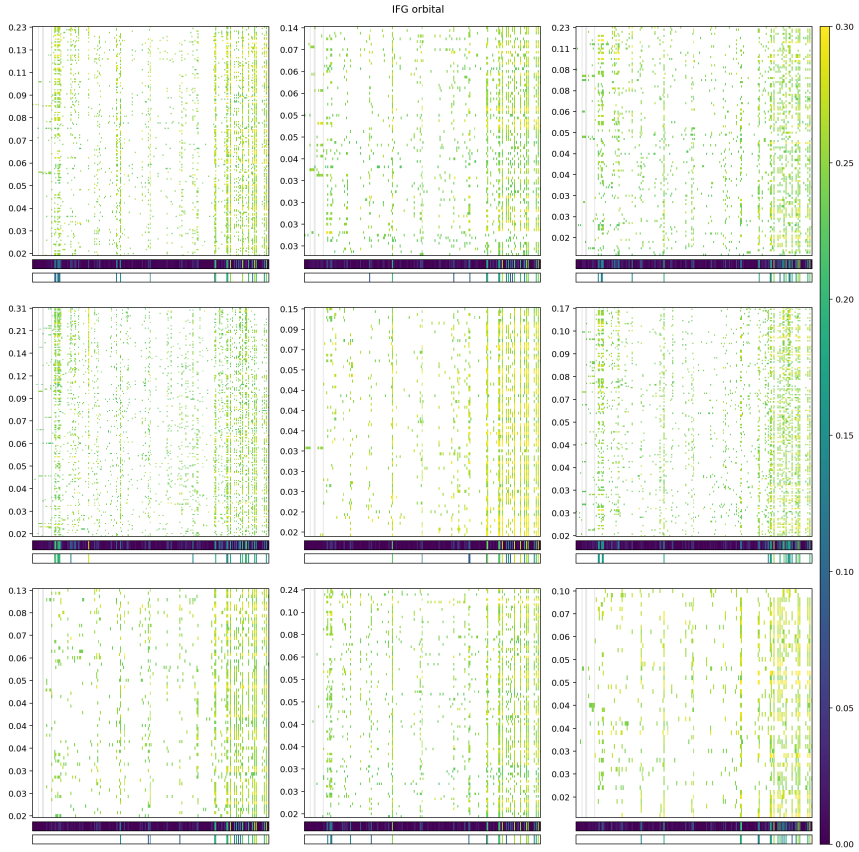


Figure E.14: Absolute task similarities of each voxel within the orbital portion of inferior frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

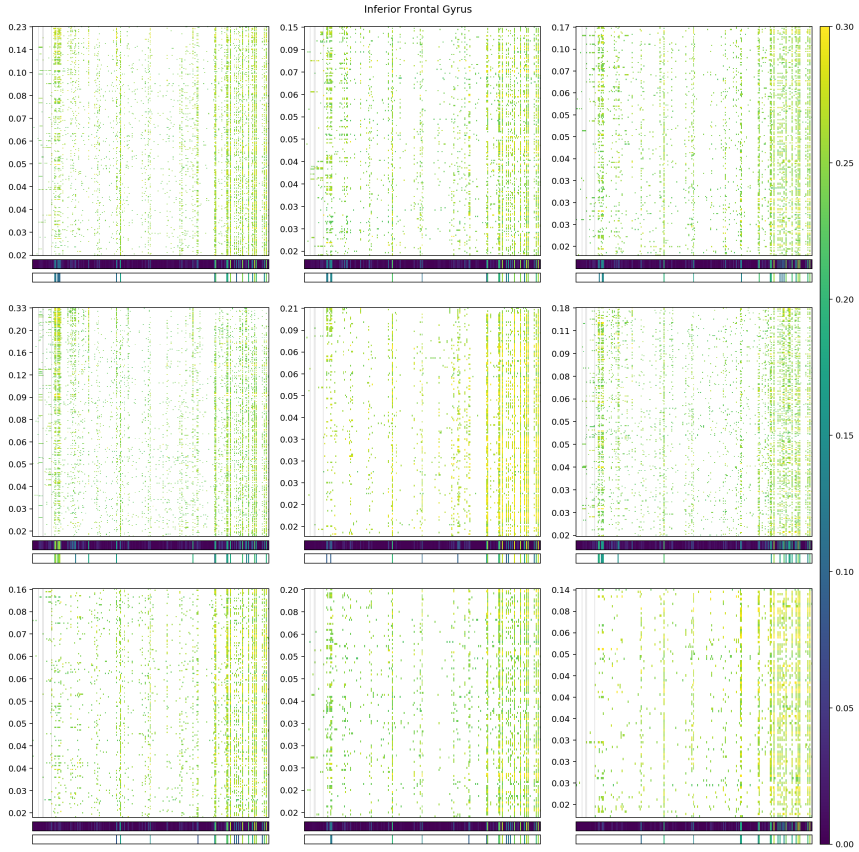


Figure E.15: Absolute task similarities of each voxel within the non-orbital part of inferior frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

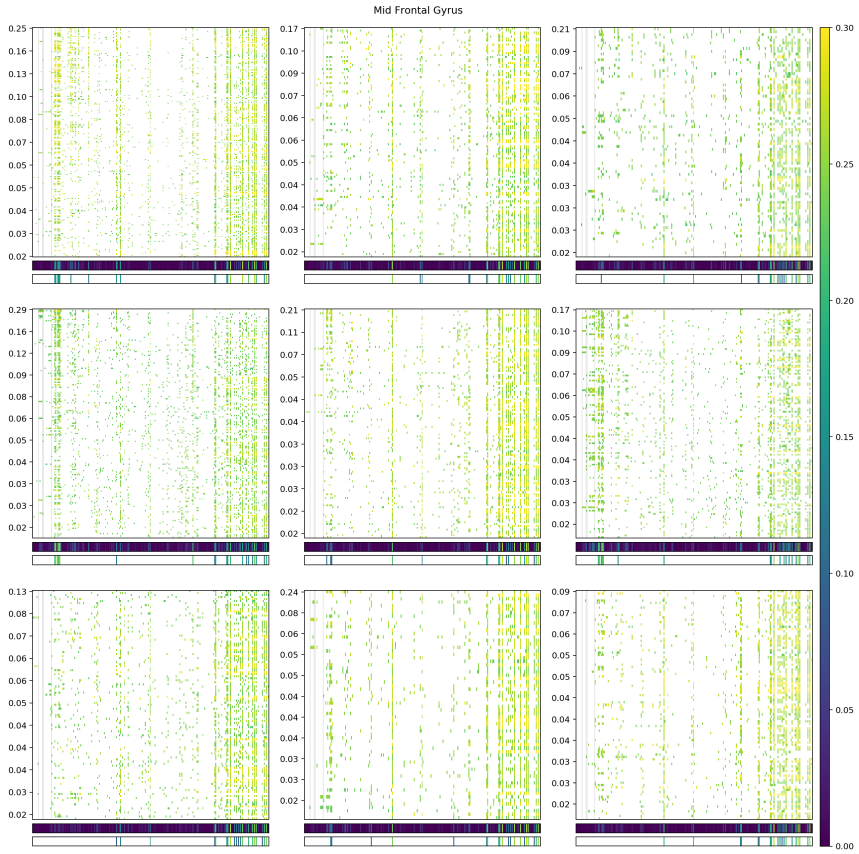


Figure E.16: Absolute task similarities of each voxel within the mid frontal gyrus as defined functionally by (Fedorenko, Hsieh, et al., 2010). Regions are projected to the participant's brain space using PyCortex (Gao et al., 2015). Each subplot is for one participant. Only voxels where the correlation between the predicted and actual values is greater than 0 according to a one-tailed t-test and corrected for false discovery rate using the Benjamini–Yekutieli procedure (Benjamini and Yekutieli, 2001) at a 0.01 level are shown. Voxels are ordered on the y-axis so that the voxels with highest correlations are at the top, with the correlations shown as the y-axis ticks. The x-axis has the predictable non-fMRI tasks/classes where classes in multi-class tasks are removed if they appear in less than 1% of examples. The heat map shows, for each voxel, the similarities of the voxels to non-fMRI classes/tasks. We filter the similarities by first removing any similarities that are not at least 2 standard erros away from 0 according to our bootstrapping procedure. Then, we keep only the 20 largest magnitude similarities for each voxel. Below each participant subplot we show a summary of the voxels for that participant computed as discussed in section 5.3.3. Below the participant's voxel summary, we show the 20 largest magnitude similarities of the summary. Vertical grey lines towards the left of each subplot dilineate 5 sections of the non-fMRI tasks: ERPs, eye-tracking, self-paced reading time, eye-tracking (again), and NLP classes/tasks. Note the smaller scale of the similarities in these plots compared to other figures. We observe that the voxel summary captures many of the similarities in the individual voxels, but does miss some structure in the data — notably some voxels are similar to ERP or reading time data, but this is not well reflected in the voxel summaries.

Bibliography

[1] Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. "Fine-grained analysis of sentence embeddings using auxiliary prediction tasks". In: *arXiv preprint arXiv:1608.04207* (2016).

[2] Ralph Adolphs, Hanna Damasio, and Daniel Tranel. "Neural systems for recognition of emotional prosody: a 3-D lesion study." In: *Emotion* 2.1 (2002), p. 23.

[3] Mark Allen, William Badecker, and Lee Osterhout. "Morphological analysis in sentence processing: An ERP study". In: *Language and Cognitive Processes* 18.4 (2003), pp. 405–430.

[4] Gerry T M Altmann. "The language machine: Psycholinguistics in review". In: *British Journal of Psychology* 92.1 (2001), pp. 129–170.

[5] Gerry T M Altmann and Yuki Kamide. "Incremental interpretation at verbs: Restricting the domain of subsequent reference". In: *Cognition* 73.3 (1999), pp. 247–264.

[6] David M Amodio and Chris D Frith. "(2006) Meeting of minds: the medial frontal cortex and social cognition". In: *Discovering the Social Mind*. Psychology Press, 2016, pp. 183–207.

[7] Sharon Ash, Peachie Moore, S Antani, G McCawley, Melissa Work, and Murray Grossman. "Trying to tell a tale: Discourse impairments in progressive aphasia and frontotemporal dementia". In: *Neurology* 66.9 (2006), pp. 1405–1413.

[8] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization". In: *arXiv preprint arXiv:1607.06450* (2016).

[9] Juliana V Baldo, Silvia A Bunge, Stephen M Wilson, and Nina F Dronkers. "Is relational reasoning dependent on language? A voxel-based lesion symptom mapping study". In: *Brain and language* 113.2 (2010), pp. 59–64.

[10] Juliana V Baldo and Nina F Dronkers. "Neural correlates of arithmetic and language comprehension: A common substrate?" In: *Neuropsychologia* 45.2 (2007), pp. 229–235.

[11] Jonathan Baxter. "A model of inductive bias learning". In: *Journal of artificial intelligence research* 12 (2000), pp. 149–198.

[12] Virginie Beaucousin, Anne Lacheret, Marie-Renée Turbelin, Michel Morel, Bernard Mazoyer, and Nathalie Tzourio-Mazoyer. "FMRI study of emotional speech comprehension". In: *Cerebral cortex* 17.2 (2007), pp. 339–352.

[13] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. "A theory of learning from different domains". In: *Machine learning* 79.1-2 (2010), pp. 151–175.

[14] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. "Analysis of representations for domain adaptation". In: *Advances in neural information processing systems*. 2007, pp. 137–144.

[15] Shai Ben-David and Reba Schuller. "Exploiting task relatedness for multiple task learning". In: *Learning Theory and Kernel Machines*. Springer, 2003, pp. 567–580.

[16] Yoav Benjamini and Yosef Hochberg. “Controlling the false discovery rate: a practical and powerful approach to multiple testing”. In: *Journal of the Royal statistical society: series B (Methodological)* 57.1 (1995), pp. 289–300.

[17] Yoav Benjamini and Daniel Yekutieli. “The control of the false discovery rate in multiple testing under dependency”. In: *Annals of statistics* (2001), pp. 1165–1188.

[18] Jos J A van Berkum, Peter Hagoort, and Colin M Brown. “Semantic integration in sentences and discourse: Evidence from the {N400}”. In: *Journal of cognitive neuroscience* 11.6 (1999), pp. 657–671.

[19] Jeffrey R Binder, Rutvik H Desai, William W Graves, and Lisa L Conant. “Where is the semantic system? A critical review and meta-analysis of 120 functional neuroimaging studies”. In: *Cerebral cortex* 19.12 (2009), pp. 2767–2796.

[20] Ina Bornkessel-Schlesewsky and Matthias Schlesewsky. “Linguistic sequence processing and the prefrontal cortex”. In: *The Open Medical Imaging Journal* 6.1 (2012).

[21] J Brennan, Y Nir, U Hasson, R Malach, D J Heeger, and L Pylkkänen. “Syntactic structure building in the anterior temporal lobe during natural story listening”. In: *Brain and language* (2010).

[22] Jonathan R Brennan, Edward P Stabler, Sarah E Van Wagenen, Wen-Ming Luh, and John T Hale. “Abstract linguistic structure correlates with temporal activity during naturalistic comprehension”. In: *Brain and language* 157 (2016), pp. 81–94.

[23] Rich Caruana. “Multitask learning”. In: *Machine learning* 28.1 (1997), pp. 41–75.

[24] Anjan Chatterjee, Lynn M Maher, LJ Gonzalez Rothi, and Kenneth M Heilman. “Asyntactic thematic role assignment: The use of a temporal-spatial strategy”. In: *Brain and Language* 49.2 (1995), pp. 125–139.

[25] Xi Chen, Seyoung Kim, Qihang Lin, Jaime G Carbonell, and Eric P Xing. “Graph-structured multi-task regression and an efficient optimization method for general fused lasso”. In: *arXiv preprint arXiv:1005.3579* (2010).

[26] Agatha Christie. *The Mysterious Affair at Styles*. Urbana, Illinois: Project Gutenberg, 1920. URL: https://www.gutenberg.org/ebooks/863.

[27] Radoslaw Martin Cichy, Dimitrios Pantazis, and Aude Oliva. “Similarity-based fusion of MEG and fMRI reveals spatio-temporal dynamics in human cortex during visual object recognition”. In: *Cerebral Cortex* 26.8 (2016), pp. 3563–3579.

[28] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. “What Does BERT Look At? An Analysis of BERT’s Attention”. In: *arXiv preprint arXiv:1906.04341* (2019).

[29] Alexis Conneau, German Kruszewski, Guillaume Lample, Lo\”\ic Barrault, and Marco Baroni. “What you can cram into a single vector: Probing sentence embeddings for linguistic properties”. In: *arXiv preprint arXiv:1805.01070* (2018).

[30] British National Consortium. *The BNC Baby*. 2005. URL: http://www.natcorp.ox.ac.uk/.

[31] Uschi Cop, Nicolas Dirix, Denis Drieghe, and Wouter Duyck. “Presenting GECO: An eyetracking corpus of monolingual and bilingual sentence reading”. In: *Behavior research methods* 49.2 (2017), pp. 602–615.

[32] H Branch Coslett. “Spatial influences on motor and language function”. In: *Neuropsychologia* 37.6 (1999), pp. 695–706.

[33] Seana Coulson, Jonathan W King, and Marta Kutas. “Expect the unexpected: Event-related brain response to morphosyntactic violations”. In: *Language and cognitive processes* 13.1 (1998), pp. 21–58.

[34] Tolga Çukur, Shinji Nishimoto, Alexander G Huth, and Jack L Gallant. "Attention during natural vision warps semantic representation across the human brain". In: *Nature neuroscience* 16.6 (2013), pp. 763–770.

[35] Stanislas Dehaene. *Reading in the brain: The new science of how we read*. Penguin, 2009.

[36] Stanislas Dehaene, Laurent Cohen, Mariano Sigman, and Fabien Vinckier. "The neural code for written words: a proposal". In: *Trends in cognitive sciences* 9.7 (2005), pp. 335–341.

[37] Barry J Devereux, Alex Clarke, Andreas Marouchos, and Lorraine K Tyler. "Representational similarity analysis reveals commonalities and differences in the semantic processing of words and objects". In: *Journal of Neuroscience* 33.48 (2013), pp. 18906–18916.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[39] David Dowty. "Thematic proto-roles and argument selection". In: *language* 67.3 (1991), pp. 547–619.

[40] Nina F Dronkers, David P Wilkins, Robert D Van Valin, Brenda B Redfern, and Jeri J Jaeger. "Lesion analysis of the brain areas involved in language comprehension". In: *Cognition* 92.1 (2004), pp. 145–177.

[41] Jeffrey L Elman. "Finding structure in time". In: *Cognitive science* 14.2 (1990), pp. 179–211.

[42] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Visualizing higher-layer features of a deep network". In: *University of Montreal* 1341.3 (2009), p. 1.

[43] Theodoros Evgeniou, Charles A Micchelli, and Massimiliano Pontil. "Learning multiple tasks with kernel methods". In: *Journal of machine learning research* 6.Apr (2005), pp. 615–637.

[44] Evelina Fedorenko, Po-Jang Hsieh, Alfonso Nieto-Castañón, Susan Whitfield-Gabrieli, and Nancy Kanwisher. "New method for fMRI investigations of language: defining ROIs functionally in individual subjects". In: *Journal of neurophysiology* 104.2 (2010), pp. 1177–1194.

[45] Evelina Fedorenko and Sharon L Thompson-Schill. "Reworking the language network". In: *Trends in cognitive sciences* 18.3 (2014), pp. 120–126.

[46] Laurie B Feldman and Darinka Andjelković. "Morphological analysis in word recognition". In: *Advances in psychology*. Vol. 94. Elsevier, 1992, pp. 343–360.

[47] Evelyn C Ferstl and D Yves von Cramon. "What does the frontomedian cortex contribute to language processing: coherence or theory of mind?" In: *Neuroimage* 17.3 (2002), pp. 1599–1612.

[48] Bruce Fischl. "FreeSurfer". In: *Neuroimage* 62.2 (2012), pp. 774–781.

[49] Ira Fischler, Donald G Childers, Teera Achariyapaopan, and Nathan W Perry Jr. "Brain potentials during sentence verification: Automatic aspects of comprehension". In: *Biological Psychology* 21.2 (1985), pp. 83–105.

[50] Jonathan R Folstein and Cyma Van Petten. "After the P3: late executive processes in stimulus categorization". In: *Psychophysiology* 48.6 (2011), pp. 825–841.

[51] Stefan L Frank, Irene Fernandez Monsalve, Robin L Thompson, and Gabriella Vigliocco. "Reading time data for evaluating broad-coverage models of English sentence processing". In: *Behavior Research Methods* 45.4 (2013), pp. 1182–1190.

[52] Stefan L Frank, Leun J Otten, Giulia Galli, and Gabriella Vigliocco. "The ERP response to the amount of information conveyed by words in sentences". In: *Brain and language* 140 (2015), pp. 1–11.

[53] Angela D Friederici. "The brain basis of language processing: from structure to function". In: *Physiological reviews* 91.4 (2011), pp. 1357–1392.

[54] Angela D Friederici. “The cortical language circuit: from auditory perception to sentence comprehension”. In: *Trends in cognitive sciences* 16.5 (2012), pp. 262–268.

[55] Angela D Friederici. “Towards a neural basis of auditory sentence processing”. In: *Trends in cognitive sciences* 6.2 (2002), pp. 78–84.

[56] Xiao Fu, Kejun Huang, Otilia Stretcu, Hyun Ah Song, Evangelos Papalexakis, Partha Talukdar, Tom Mitchell, Nicholas Sidiropoulo, Christos Faloutsos, and Barnabas Poczos. “BRAINZOOM: High resolution reconstruction from multi-modal brain signals”. In: *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM. 2017, pp. 216–227.

[57] Alona Fyshe, Partha P Talukdar, Brian Murphy, and Tom M Mitchell. “Interpretable semantic vectors from a joint model of brain-and text-based meaning”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Vol. 1. 2014, pp. 489–499.

[58] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. “Domain-adversarial training of neural networks”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2096–2030.

[59] James S Gao, Alexander G Huth, Mark D Lescroart, and Jack L Gallant. “Pycortex: an interactive surface visualizer for fMRI”. In: *Frontiers in neuroinformatics* 9 (2015), p. 23.

[60] Marie St George and Suzanne Mannes. “Global semantic expectancy and language comprehension”. In: *Journal of cognitive neuroscience* 6.1 (1994), pp. 70–83.

[61] Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. “A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers”. In: *International conference on machine learning*. 2013, pp. 738–746.

[62] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[63] R F Goldberg, C A Perfetti, and W Schneider. “Distinct and common cortical activations for multimodal semantic categories”. In: *Cognitive, Affective, & Behavioral Neuroscience* 6.3 (2006), pp. 214–222.

[64] Laura M Gonnerman, Mark S Seidenberg, and Elaine S Andersen. “Graded semantic and phonological similarity effects in priming: Evidence for a distributed connectionist approach to morphology.” In: *Journal of experimental psychology: General* 136.2 (2007), p. 323.

[65] Umut Güçlü and Marcel AJ van Gerven. “Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream”. In: *Journal of Neuroscience* 35.27 (2015), pp. 10005–10014.

[66] Thomas C Gunter, Laurie A Stowe, and Gusbertus Mulder. “When syntax meets semantics”. In: *Psychophysiology* 34.6 (1997), pp. 660–676.

[67] Peter Hagoort, Lea Hald, Marcel Bastiaansen, and Karl Magnus Petersson. “Integration of word meaning and world knowledge in language comprehension”. In: *science* 304.5669 (2004), pp. 438–441.

[68] Peter Hagoort, Marlies Wassenaar, and Colin Brown. “Real-time semantic compensation in patients with agrammatic comprehension: Electrophysiological evidence for multiple-route plasticity”. In: *Proceedings of the National Academy of Sciences* 100.7 (2003), pp. 4340–4345.

[69] Peter Hagoort, Marlies Wassenaar, and Colin M Brown. “Syntax-related ERP-effects in Dutch”. In: *Cognitive Brain Research* 16.1 (2003), pp. 38–50.

[70] Anja Hahne and Angela D Friederici. “Differential task effects on semantic and syntactic processes as revealed by ERPs”. In: *Cognitive Brain Research* 13.3 (2002), pp. 339–356.

[71] Anja Hahne and Angela D Friederici. "Electrophysiological evidence for two steps in syntactic analysis: Early automatic and late controlled processes". In: *Journal of cognitive neuroscience* 11.2 (1999), pp. 194–205.

[72] John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan R Brennan. "Finding Syntax in Human Encephalography with Beam Search". In: *arXiv preprint arXiv:1806.04127* (2018).

[73] Bin He, Abbas Sohrabpour, Emery Brown, and Zhongming Liu. "Electrophysiological source imaging: a noninvasive window to brain dynamics". In: *Annual review of biomedical engineering* 20 (2018), pp. 171–196.

[74] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[75] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. "Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals". In: *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics. 2010, pp. 33–38.

[76] Dan Hendrycks and Kevin Gimpel. "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415* (2016).

[77] Isabelle Hesling, Sylvain Clément, Martine Bordessoules, and Michele Allard. "Cerebral mechanisms of prosodic integration: evidence from connected speech". In: *Neuroimage* 24.4 (2005), pp. 937–947.

[78] Gregory Hickok and David Poeppel. "The cortical organization of speech processing". In: *Nature Reviews Neuroscience* 8.5 (2007), p. 393.

[79] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[80] Paul Hoffman, Elizabeth Jefferies, and Matthew A Lambon Ralph. "Ventrolateral prefrontal cortex plays an executive regulation role in comprehension of abstract words: convergent neuropsychological and repetitive TMS evidence". In: *Journal of Neuroscience* 30.46 (2010), pp. 15450–15456.

[81] Nora Hollenstein, Antonio de la Torre, Nicolas Langer, and Ce Zhang. "CogniVal: A Framework for Cognitive Word Embedding Evaluation". In: *arXiv preprint arXiv:1909.09001* (2019).

[82] Jeremy Howard and Sebastian Ruder. "Fine-tuned Language Models for Text Classification". In: *arXiv preprint arXiv:1801.06146* (2018).

[83] Falk Huettig. "Four central questions about prediction in language processing". In: *Brain Research* 1626 (2015), pp. 118–135.

[84] Alexander G Huth, Wendy A de Heer, Thomas L Griffiths, Frédéric E Theunissen, and Jack L Gallant. "Natural speech reveals the semantic maps that tile human cerebral cortex". In: *Nature* 532.7600 (2016), pp. 453–458.

[85] Laurent Jacob, Jean-philippe Vert, and Francis R. Bach. "Clustered Multi-Task Learning: A Convex Formulation". In: *Advances in Neural Information Processing Systems 21*. Ed. by D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou. Curran Associates, Inc., 2009, pp. 745–752. URL: http://papers.nips.cc/paper/3499-clustered-multi-task-learning-a-convex-formulation.pdf.

[86] Shailee Jain and Alexander Huth. "Incorporating Context into Language Encoding Models for fMRI". In: *Advances in Neural Information Processing Systems*. Cold Spring Harbor Laboratory, 2018, p. 327601.

[87] J A John and Norman R Draper. “An alternative family of transformations”. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 29.2 (1980), pp. 190–197.

[88] Marcel A Just and Patricia A Carpenter. “A theory of reading: From eye fixations to comprehension.” In: *Psychological review* 87.4 (1980), p. 329.

[89] Marcel A Just, Patricia A Carpenter, and Jacqueline D Woolley. “Paradigms and processes in reading comprehension.” In: *Journal of experimental psychology: General* 111.2 (1982), p. 228.

[90] Kendrick N Kay, Thomas Naselaris, Ryan J Prenger, and Jack L Gallant. “Identifying natural images from human brain activity”. In: *Nature* 452.7185 (2008), p. 352.

[91] Alexander JE Kell, Daniel LK Yamins, Erica N Shook, Sam V Norman-Haignere, and Josh H McDermott. “A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy”. In: *Neuron* 98.3 (2018), pp. 630–644.

[92] David Kemmerer. *Cognitive neuroscience of language*. Psychology Press, 2014.

[93] Alan Kennedy, Robin Hill, and Joël Pynte. “The dundee corpus”. In: *Proceedings of the 12th European conference on eye movement*. 2003.

[94] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. “Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context”. In: *arXiv preprint arXiv:1805.04623* (2018).

[95] Seyoung Kim and Eric P Xing. “Tree-guided group lasso for multi-task regression with structured sparsity.” In: *ICML*. Vol. 2. Citeseer. 2010, p. 1.

[96] Jonathan W King and Marta Kutas. “Who did what and when? Using word-and clause-level ERPs to monitor working memory usage in reading”. In: *Journal of cognitive neuroscience* 7.3 (1995), pp. 376–395.

[97] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[98] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. “Representational similarity analysis-connecting the branches of systems neuroscience”. In: *Frontiers in systems neuroscience* 2 (2008), p. 4.

[99] Gina R Kuperberg. “Neural mechanisms of language comprehension: Challenges to syntax”. In: *Brain research* 1146 (2007), pp. 23–49.

[100] Gina R Kuperberg, Tatiana Sitnikova, David Caplan, and Phillip J Holcomb. “Electrophysiological distinctions in processing conceptual relationships within simple sentences”. In: *Cognitive Brain Research* 17.1 (2003), pp. 117–129.

[101] Marta Kutas. “In the company of other words: Electrophysiological evidence for single-word and sentence context effects”. In: *Language and cognitive processes* 8.4 (1993), pp. 533–572.

[102] Marta Kutas and Kara D Federmeier. “Thirty years and counting: finding meaning in the N400 component of the event-related brain potential ({ERP})”. In: *Annual review of psychology* 62 (2011), pp. 621–647.

[103] Marta Kutas and Steven A Hillyard. “Brain potentials during reading reflect word expectancy and semantic association”. In: *Nature* 307.5947 (1984), p. 161.

[104] Marta Kutas, Cyma K Van Petten, and Robert Kluender. “Psycholinguistics electrified II (1994–2005)”. In: *Handbook of Psycholinguistics (Second Edition)*. Elsevier, 2006, pp. 659–724.

[105] Matthew A Lambon Ralph, Karen Sage, Roy W Jones, and Emily J Mayberry. “Coherent concepts are computed in the anterior temporal lobes”. In: *Proceedings of the National Academy of Sciences* 107.6 (2010), pp. 2717–2722.

[106] Sarah Laszlo and David C Plaut. “A neurally plausible parallel distributed processing model of event-related potential word reading data”. In: *Brain and language* 120.3 (2012), pp. 271–281.

[107] Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. "Assessing the ability of LSTMs to learn syntax-sensitive dependencies". In: *arXiv preprint arXiv:1611.01368* (2016).

[108] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. "Domain adaptation: Learning bounds and algorithms". In: *arXiv preprint arXiv:0902.3430* (2009).

[109] James L McClelland and Jeffrey L Elman. "The TRACE model of speech perception". In: *Cognitive psychology* 18.1 (1986), pp. 1–86.

[110] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. "Regularizing and optimizing {LSTM} language models". In: *arXiv preprint arXiv:1708.02182* (2017).

[111] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. "Pointer sentinel mixture models". In: *arXiv preprint arXiv:1609.07843* (2016).

[112] Tom M Mitchell, Svetlana V Shinkareva, Andrew Carlson, Kai-Min Chang, Vicente L Malave, Robert A Mason, and Marcel Adam Just. "Predicting human brain activity associated with the meanings of nouns". In: *science* 320.5880 (2008), pp. 1191–1195.

[113] Brian Murphy, Partha Talukdar, and Tom Mitchell. "Selecting corpus-semantic models for neurolinguistic decoding". In: *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics. 2012, pp. 114–123.

[114] S Nishimoto, A T Vu, T Naselaris, Y Benjamini, B Yu, and J L Gallant. "Reconstructing visual experiences from brain activity evoked by natural movies". In: *Current Biology* (2011).

[115] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. "Feature Visualization". In: *Distill* (2017). https://distill.pub/2017/feature-visualization. DOI: 10.23915/distill.00007.

[116] Dirk-Bart den Ouden, Steve Fix, Todd B Parrish, and Cynthia K Thompson. "Argument structure effects in action verb naming in static and dynamic conditions". In: *Journal of Neurolinguistics* 22.2 (2009), pp. 196–215.

[117] Mark M Palatucci. "Thought recognition: predicting and decoding brain activity using the zero-shot learning model". In: (2011).

[118] Christophe Pallier, Anne-Dominique Devauchelle, and Stanislas Dehaene. "Cortical representation of the constituent structure of sentences". In: *Proceedings of the National Academy of Sciences* 108.6 (2011), pp. 2522–2527.

[119] Martha Palmer, Daniel Gildea, and Paul Kingsbury. "The proposition bank: An annotated corpus of semantic roles". In: *Computational linguistics* 31.1 (2005), pp. 71–106.

[120] Maria Palolahti, Sakari Leino, Markus Jokela, Kreetta Kopra, and Petri Paavilainen. "Event-related potentials suggest early interaction between syntax and semantics during on-line sentence comprehension". In: *Neuroscience Letters* 384.3 (2005), pp. 222–227.

[121] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in PyTorch". In: *NIPS-W*. 2017.

[122] Francisco Pereira, Bin Lou, Brianna Pritchett, Samuel Ritter, Samuel J Gershman, Nancy Kanwisher, Matthew Botvinick, and Evelina Fedorenko. "Toward a universal decoder of linguistic meaning from brain activation". In: *Nature communications* 9.1 (2018), p. 963.

[123] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. "Deep contextualized word representations". In: *Proc. of NAACL*. 2018.

[124] David C Plaut and Marlene Behrmann. "Complementary neural representations for faces and words: A computational exploration". In: *Cognitive neuropsychology* 28.3-4 (2011), pp. 251–275.

[125] David C Plaut, James L McClelland, Mark S Seidenberg, and Karalyn Patterson. "Understanding normal and impaired word reading: computational principles in quasi-regular domains." In: *Psychological review* 103.1 (1996), p. 56.

[126] Chantel S Prat and Marcel Adam Just. "Exploring the neural dynamics underpinning individual differences in sentence comprehension". In: *Cerebral cortex* 21.8 (2011), pp. 1747–1760.

[127] Altaf Rahman and Vincent Ng. "Resolving complex cases of definite pronouns: the winograd schema challenge". In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics. 2012, pp. 777–789.

[128] Kathleen Rastle, Matthew H Davis, and Boris New. "The broth in my brother's brothel: Morpho-orthographic segmentation in visual word recognition". In: *Psychonomic Bulletin & Review* 11.6 (2004), pp. 1090–1098.

[129] Keith Rayner. "Eye movements and attention in reading, scene perception, and visual search". In: *The quarterly journal of experimental psychology* 62.8 (2009), pp. 1457–1506.

[130] Keith Rayner. "Eye movements in reading and information processing: 20 years of research." In: *Psychological bulletin* 124.3 (1998), p. 372.

[131] Keith Rayner and Alexander Pollatsek. "Eye-movement control in reading". In: *Handbook of Psycholinguistics (Second Edition)*. Elsevier, 2006, pp. 613–657.

[132] Erik D Reichle, Keith Rayner, and Alexander Pollatsek. "The EZ Reader model of eye-movement control in reading: Comparisons to other models". In: *Behavioral and brain sciences* 26.4 (2003), pp. 445–476.

[133] Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. "Semantic proto-roles". In: *Transactions of the Association for Computational Linguistics* 3 (2015), pp. 475–488.

[134] Maximilian Riesenhuber and Tomaso Poggio. "Hierarchical models of object recognition in cortex". In: *Nature neuroscience* 2.11 (1999), pp. 1019–1025.

[135] Corianne Rogalsky, William Matchin, and Gregory Hickok. "Broca's area, sentence comprehension, and working memory: an fMRI study". In: *Frontiers in human neuroscience* 2 (2008), p. 14.

[136] J K Rowling. *Harry Potter and the Sorcerer's Stone*. 1st ed. Vol. 1. New York: Scholastic, June 1999.

[137] David E Rumelhart, James L McClelland, P D P Research Group, et al. *Parallel distributed processing*. Vol. 1. MIT press Cambridge, MA, 1987.

[138] Avishek Saha, Piyush Rai, Hal DaumÃ, Suresh Venkatasubramanian, et al. "Online learning of multiple tasks and their relationships". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 2011, pp. 643–651.

[139] Ned T Sahin, Steven Pinker, Sydney S Cash, Donald Schomer, and Eric Halgren. "Sequential processing of lexical, grammatical, and phonological information within Broca's area". In: *Science* 326.5951 (2009), pp. 445–449.

[140] Ned T Sahin, Steven Pinker, and Eric Halgren. "Abstract grammatical processing of nouns and verbs in Broca's area: evidence from fMRI". In: *Cortex* 42.4 (2006), pp. 540–562.

[141] Rebecca Saxe and Nancy Kanwisher. "People thinking about thinking people: the role of the temporo-parietal junction in "theory of mind"". In: *Neuroimage* 19.4 (2003), pp. 1835–1842.

[142] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Kailyn Schmidt, et al. "Brain-score: Which artificial neural network for object recognition is most brain-like?" In: *BioRxiv* (2018), p. 407007.

[143] Dan Schwartz and Tom Mitchell. “Understanding language-elicited EEG data by predicting it from a fine-tuned language model”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (2019), pp. 43–57.

[144] Dan Schwartz, Mariya Toneva, and Leila Wehbe. “Inducing brain-relevant bias in natural language processing models”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 14100–14110.

[145] Elisabet Service, Päivi Helenius, Sini Maury, and Riitta Salmelin. “Localization of syntactic and semantic brain responses using magnetoencephalography”. In: *Journal of Cognitive Neuroscience* 19.7 (2007), pp. 1193–1205.

[146] Roger N Shepard. “Stimulus and response generalization: A stochastic model relating generalization to distance in psychological space”. In: *Psychometrika* 22.4 (1957), pp. 325–345.

[147] Natalia Silveira, Timothy Dozat, Marie-Catherine De Marneffe, Samuel R Bowman, Miriam Connor, John Bauer, and Christopher D Manning. “A Gold Standard Dependency Corpus for English.” In: *LREC*. Citeseer. 2014, pp. 2897–2904.

[148] W Kyle Simmons, Vimal Ramjee, Michael S Beauchamp, Ken McRae, Alex Martin, and Lawrence W Barsalou. “A common neural substrate for perceiving and knowing about color”. In: *Neuropsychologia* 45.12 (2007), pp. 2802–2810.

[149] Tineke M Snijders, Theo Vosse, Gerard Kempen, Jos J A Van Berkum, Karl Magnus Petersson, and Peter Hagoort. “Retrieval and unification of syntactic structure in sentence comprehension: an f{MRI} study using word-category ambiguity”. In: *Cerebral cortex* 19.7 (2008), pp. 1493–1503.

[150] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. “Recursive deep models for semantic compositionality over a sentiment treebank”. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, pp. 1631–1642.

[151] Nicole K Speer, Jeffrey M Zacks, and Jeremy R Reynolds. “Human brain activity time-locked to narrative event boundaries”. In: *Psychological Science* 18.5 (2007), pp. 449–455.

[152] Karsten Steinhauer and John E Drury. “On the early left-anterior negativity (ELAN) in syntax studies”. In: *Brain and language* 120.2 (2012), pp. 135–162.

[153] Karsten Steinhauer and Angela D Friederici. “Prosodic boundaries, comma rules, and brain responses: The closure positive shift in ERPs as a universal marker for prosodic phrasing in listeners and readers”. In: *Journal of psycholinguistic research* 30.3 (2001), pp. 267–295.

[154] Samu Taulu, Matti Kajola, and Juha Simola. “Suppression of interference and artifacts by the signal space separation method”. In: *Brain topography* 16.4 (2004), pp. 269–275.

[155] Samu Taulu and Juha Simola. “Spatiotemporal signal space separation method for rejecting nearby interference in MEG measurements”. In: *Physics in Medicine & Biology* 51.7 (2006), p. 1759.

[156] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. “What do you learn from context? probing for sentence structure in contextualized word representations”. In: *arXiv preprint arXiv:1905.06316* (2019).

[157] Cynthia K Thompson, Borna Bonakdarpour, Stephen C Fix, Henrike K Blumenfeld, Todd B Parrish, Darren R Gitelman, and M-Marsel Mesulam. “Neural correlates of verb argument structure processing”. In: *Journal of Cognitive Neuroscience* 19.11 (2007), pp. 1753–1767.

[158] Dianne E Thornhill and Cyma Van Petten. “Lexical versus conceptual anticipation during sentence processing: Frontal positivity and N400 ERP components”. In: *International Journal of Psychophysiology* 83.3 (2012), pp. 382–392.

[159] Malathi Thothathiri, Daniel Y Kimberg, and Myrna F Schwartz. “The neural basis of reversible sentence comprehension: evidence from voxel-based lesion symptom mapping in aphasia”. In: *Journal of cognitive neuroscience* 24.1 (2012), pp. 212–222.

[160] Sebastian Thrun and Joseph O’Sullivan. “Discovering structure in multiple learning tasks: The TC algorithm”. In: *ICML*. Vol. 96. 1996, pp. 489–497.

[161] Mariya Toneva and Leila Wehbe. “Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain)”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 14928–14938.

[162] Matthew Traxler and Morton Ann Gernsbacher. *Handbook of psycholinguistics*. Elsevier, 2011.

[163] Vanessa Troiani, Maria A Fernández-Seara, Ze Wang, John A Detre, Sherry Ash, and Murray Grossman. “Narrative speech production: an fMRI study using continuous arterial spin labeling”. In: *Neuroimage* 40.2 (2008), pp. 932–939.

[164] Lorraine K Tyler, William D Marslen-Wilson, Billi Randall, Paul Wright, Barry J Devereux, Jie Zhuang, Marina Papoutsi, and Emmanuel A Stamatakis. “Left inferior frontal cortex and syntax: function, structure and behaviour in patients with left hemisphere damage”. In: *Brain* 134.2 (2011), pp. 415–431.

[165] Michael T Ullman, Roumyana Pancheva, Tracy Love, Eiling Yee, David Swinney, and Gregory Hickok. “Neural correlates of lexicon and grammar: Evidence from the production, reading, and judgment of inflection in aphasia”. In: *Brain and Language* 93.2 (2005), pp. 185–238.

[166] Cyma Van Petten. “A comparison of lexical and sentence-level context effects in event-related potentials”. In: *Language and Cognitive Processes* 8.4 (1993), pp. 485–531.

[167] Cyma Van Petten and Marta Kutas. “Interactions between sentence context and word frequency in event-related brain potentials”. In: *Memory & cognition* 18.4 (1990), pp. 380–393.

[168] Cyma Van Petten and Barbara J Luka. “Neural localization of semantic context effects in electromagnetic and hemodynamic studies”. In: *Brain and language* 97.3 (2006), pp. 279–293.

[169] Cyma Van Petten and Barbara J Luka. “Prediction during language comprehension: Benefits, costs, and ERP components”. In: *International Journal of Psychophysiology* 83.2 (2012), pp. 176–190.

[170] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention is all you need”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 5998–6008.

[171] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. “SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 3266–3280. URL: http://papers.nips.cc/paper/8589-superglue-a-stickier-benchmark-for-general-purpose-language-understanding-systems.pdf.

[172] Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. “Glue: A multi-task benchmark and analysis platform for natural language understanding”. In: *arXiv preprint arXiv:1804.07461* (2018).

[173] Aria Wang, Michael Tarr, and Leila Wehbe. "Neural Taskonomy: Inferring the Similarity of Task-Derived Representations from Brain Activity". In: *Advances in Neural Information Processing Systems*. 2019, pp. 15475–15485.

[174] Jing Wang, Julie A Conder, David N Blitzer, and Svetlana V Shinkareva. "Neural representation of abstract and concrete concepts: A meta-analysis of neuroimaging studies". In: *Human brain mapping* 31.10 (2010), pp. 1459–1468.

[175] Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. "Balancing Training for Multilingual Neural Machine Translation". In: *arXiv preprint arXiv:2004.06748* (2020).

[176] Leila Wehbe, Brian Murphy, Partha Talukdar, Alona Fyshe, Aaditya Ramdas, and Tom Mitchell. "Simultaneously Uncovering the Patterns of Brain Regions Involved in Different Story Reading Subprocesses". In: *PLOS ONE* 9.11 (Nov. 2014). Ed. by Kevin Paterson, e112575. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0112575. URL: http://dx.plos.org/10.1371/journal.pone.0112575.

[177] Leila Wehbe, Ashish Vaswani, Kevin Knight, and Tom Mitchell. "Aligning context-based statistical models of language with brain activity during reading". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 233–243. URL: http://www.aclweb.org/anthology/D14-1030.

[178] Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. "OntoNotes Release 4.0". In: *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium* (2011).

[179] Aaron Steven White, Pushpendre Rastogi, Kevin Duh, and Benjamin Van Durme. "Inference is everything: Recasting semantic resources into a unified evaluation framework". In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2017, pp. 996–1005.

[180] Aaron Steven White, Kyle Rawlins, and Benjamin Van Durme. "The semantic proto-role linking model". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. 2017, pp. 92–98.

[181] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8.3-4 (1992), pp. 229–256.

[182] Jurriaan Witteman, Vincent JP Van Heuven, and Niels O Schiller. "Hearing feelings: a quantitative meta-analysis on the neuroimaging literature of emotional prosody perception". In: *Neuropsychologia* 50.12 (2012), pp. 2752–2763.

[183] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. "Performance-optimized hierarchical models predict neural responses in higher visual cortex". In: *Proceedings of the National Academy of Sciences* 111.23 (2014), pp. 8619–8624.

[184] Yu Zhang and Dit-Yan Yeung. "A convex formulation for learning task relationships in multi-task learning". In: *arXiv preprint arXiv:1203.3536* (2012).

[185] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 19–27.