Real-time decoding of perceived and produced speech from human cortical activity

by

David Moses

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Bioengineering

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, SAN FRANCISCO

[redacted]

Acknowledgments

First and foremost, I want to thank my mother Carol and father Moshe for their unwavering support throughout my entire life, including my graduate studies. Ever since I was a child, they have always placed a high priority on my education and have always encouraged challenging but rewarding pursuits of knowledge. I hope that receiving this doctorate degree can at least partially honor their commitment to foster my intellectual growth throughout my upbringing and the sacrifices that they have made on my behalf.

I also want to thank my brother Daniel, who has been an irreplaceable friend to me over the years. Even though we lived in different cities during my graduate studies, he consistently made time to chat and play online games whenever I wanted to unwind after particularly stressful days. These sessions often made a world of difference for me, and it was extremely comforting to know that he would support me in whatever ways he could despite the distance.

A big thanks goes out to my research advisor and committee chair, Dr. Edward Chang. Before joining his lab, I struggled to determine what kind of research I wanted to do during my graduate studies. During my time in his lab, he was a constant source of motivation and guidance, steering me towards projects that he knew I would excel at. I am indebted to him for all of the opportunities and support that he has provided me throughout my time in his lab.

I would also like to thank my other committee members, Dr. Jack Gallant, Dr. Nelson Morgan, and Dr. Kristofer Bouchard, for their useful feedback and guidance as well as their availability despite their busy schedules.

I also want to thank all of the other members of Dr. Chang's lab for making my workplace friendly and intellectually stimulating. I will cherish all of the interesting and sometimes polarizing conversations that we had during our lunch outings. A special thanks goes out to Dr. Matthew Leonard, who has been a limitless source of wisdom and mentorship ever since the first day of my rotation in the lab.

I also want to acknowledge and thank all of the patients who volunteered to participate in all of the research I have been a part of since joining the lab. Despite being in an uncomfortable and stressful situation, these patients gave their time to participate in our (often repetitive) tasks out of pure kindness and desire to contribute to the collection of human scientific knowledge.

Ever since moving to California to start graduate school, I have made truly special and incredible friends that I am very grateful for. However, I must give an extremely special shoutout to the other members of a friend group known only as *The Crunkmen*: Charles (Travis) Howell, Jesus Cortez, Ryan Oringer, Steven Boswell, and Stephen Haff. It was one of the greatest fortunes of my time in undergraduate school to become friends with these fantastic individuals, and I cannot imagine my time in graduate school without their continued friendship. I also want to thank Ahsan Niazi for being a close friend and source of motivation ever since we first met in high school.

In addition to these people who I have relied on for personal and professional support, I would also like to thank the creators and maintainers of various open source software packages that I have relied on throughout my graduate career, including *Python*, *Ubuntu*, *LaTeX*, *Inkscape*, and *LibreOffice*. Finally, I would like to thank the various musical artists whose works have been mentally and emotionally stimulating during my time in graduate school, including Steven Wilson, Jim Grey, Maynard James Keenan, and Nicholas Thorburn.

Note from research advisor concerning previously published materials

This thesis contains material from the following publications:

D. A. Moses, N. Mesgarani, M. K. Leonard, and E. F. Chang, 2016. Neural speech recognition: Continuous phoneme decoding using spatiotemporal representations of human cortical activity. *Journal of Neural Engineering*, 13(5):056004. doi: 10.1088/1741-2560/13/5/056004.

D. A. Moses, M. K. Leonard, and E. F. Chang, 2018. Real-time classification of auditory sentences using evoked cortical activity in humans. *Journal of Neural Engineering*, 15(3). doi: 10.1088/1741-2552/aaab6f.

This thesis is comparable a standard thesis because the chapters represent a sequential progression of closely related research topics. David performed the majority of the work for the projects described in each chapter.

Real-time decoding of perceived and produced speech from human cortical activity

David A. Moses

Recent research has explored the functional role of the human auditory and sensorimotor cortices in perceiving and producing speech. One key finding from these studies was the characterization of how phonetic features and phonemes, which are fundamental units of speech, are represented in the brain. In this thesis, I examine how these neural representations of speech can be used as the basis for decoding what individuals hear and say in real-time. This work leverages recent advances in human neurophysiology with epilepsy patients that have high-density electrocorticography arrays implanted on the surface of their brains as part of their clinical treatment, enabling collection of cortical activity at unprecedented spatiotemporal resolutions. Using these signals, I show that spatiotemporal feature vectors containing cortical activity in the high gamma frequency band can be used to decode speech sounds that listeners perceive by employing techniques from automatic speech recognition, contributing to an emerging field of research referred to as neural speech recognition (NSR). Next, I design and evaluate a real-time system capable of reliably classifying aurally presented sentences using phoneme-level models and spatiotemporal high gamma features. Finally, I demonstrate state-of-the-art real-time decoding of perceived and produced words and sentences in a naturalistic question-and-answer paradigm, illustrating the utility of NSR in a real-world interactive application. In addition to characterizing properties of speech that can be decoded from the human brain in real-time, these findings have practical implications for the design of speech neuroprostheses to aid patients who are unable to communicate due to paralysis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Speech is one of the most complicated behaviors that humans naturally and effortlessly engage in on a daily basis. Even without considering the intricate coordination of movements required to produce speech, our ability to rapidly and robustly process acoustic speech input to extract semantic meanings is remarkable. During this biological process of speech perception, an acoustic speech waveform evokes neuronal firing patterns in cochlear hair cells that convey information through the central nervous system, eventually reaching the auditory cortex (Brugge, 1992). Although a complete description of acoustic speech processing in the human auditory cortex is not available, many studies have characterized aspects of various steps in the processing pathway. Broadly, information entering the auditory cortex is transmitted neurally to neuronal populations in the primary auditory cortex (A1) that encode low-level acoustic features, such as frequency and sound intensity (Linden and Schreiner, 2003; Moerel et al., 2014). Then, information flows from A1 to other parts of the superior temporal gyrus (STG) that are known to encode higher-level speech-related aspects of the acoustics, such as phonetic and spectrotemporal features (Hickok and Poeppel, 2007). Afterwards, lexical and semantic processing occurs in the STG, in nearby cortical areas (including the middle temporal gyrus and superior temporal sulcus), and in many other brain regions (Hickok and Poeppel, 2007; Mitchell et al., 2008; Huth et al., 2012).

The theory that the STG performs speech-related processing of acoustic inputs has evolved from research findings during the recent decades. It has been shown that cortical stimulation to the posterior STG during perception of speech sounds and tones can interrupt speech processing without significantly affecting tone discrimination (Boatman et al., 1997). Other studies have shown that the STG is more significantly modulated by speech sounds, such as syllables or words, than non-speech sounds, such as tones or even unintelligible nonwords that resembled speech (Binder et al., 2000; Canolty et al., 2007; Cibelli et al., 2015). During investigations of how speech information is encoded in the STG, the activity in this and neighboring brain regions has been shown to correlate with many high-level speech features, including lexical statistics (Cibelli et al., 2015), speech-related spectrotemporal fluctuations (Pasley et al., 2012), speaker-normalized pitch (Tang et al., 2017), and sentence onsets (Hamilton et al., 2018). Among the various types of speech processing in the STG, the type that is arguably the most fundamental to understanding speech is the encoding of phonetic features throughout this region. Phonetic features describe the ways in which speech sounds, including descriptions of the manner of articulation (for example, plosive vs. fricative) and place of articulation (for example, labial vs. dental) (Chomsky and Halle, 1968). At a more abstract level, combinations of certain phonetic features result in phonemes, which are the smallest contrastive speech units that can alter the meaning of a word, such as the /m/ sound in “mat” or the /p/ sound in “pat”. From brain activity in the STG, researchers have found categorical encoding of phoneme categories (Chang et al., 2010), localized and distributed phonetic feature encoding (Mesgarani et al., 2014), and phonotactic information encoding (Leonard et al., 2016). Overall, these results suggest that the human STG naturally extracts phonetic information from acoustic speech signals to facilitate speech comprehension.

Interestingly, many successful applications in the field of automatic speech recognition

(ASR) also use phonemes as an intermediate step during conversion of acoustic signals into words and sentences (BenZeghiba et al., 2007; Hinton et al., 2012; Kurian, 2014). In many of these applications, large datasets of acoustic speech signals were transcribed phonetically and used to train probabilistic models to yield phoneme likelihoods given unseen acoustic samples. The time series of predicted phoneme likelihoods could then be used with pre-trained language models (which describe the probabilities of observing certain speech elements given the previous elements in the sequence) to decode words and sentences. If a system was capable of inferring phoneme likelihoods from neural signals, the remainder of this decoding pipelining could be adapted to decode speech from brain activity.

In **chapter 2**, I describe the application of ASR techniques to decode phoneme sequences from brain activity and identification of effective neural features to use during decoding (Moses et al., 2016). In this work, human participants listened to aurally presented sentences while neural activity was collected from high-density electrocorticography (ECoG) grids. These ECoG grids, which were implanted over the cortical surface of epilepsy patients to aid clinicians in localizing seizure foci, enabled acquisition of neural activity at high spatial and temporal resolutions (Ball et al., 2009). The recordings sampled from multiple speech-related cortical areas, including the STG.

A major goal of this work was to determine how to extract relevant features from the collected ECoG activity that could be used during phoneme decoding. Previous studies with ECoG have identified that power in the high gamma frequency band (70–150 Hz) was strongly correlated with multi-unit spiking patterns (Crone et al., 1998) and neuronal firing synchrony (Ray et al., 2008). Furthermore, high gamma activity from the STG is significantly modulated by the presence of acoustic speech stimuli (Pasley et al., 2012; Mesgarani et al., 2014). However, it is known that auditory speech stimuli can evoke cortical responses that are both spatially complex (varying across electrodes) and temporally complex (varying in time) (Engineer et al., 2008; Buonomano and Maass, 2009). In this work I decided to use high gamma activity to represent the neural data, but I also investigated how using spatiotemporal feature vectors of high gamma activity would affect the ability to decode phonemes at each time point. After training and testing phoneme likelihood estimation models with either purely spatial or spatiotemporal features, I observed significantly higher phoneme decoding accuracies when using spatiotemporal features as opposed to spatial features, confirming that explicitly modeling the spatiotemporal dynamics of neural activity can be beneficial during discrimination tasks. This finding also resembles an approach used to improve performance in some ASR systems that involves including information about how the feature vectors change over time in the feature vectors themselves (Gold et al., 2011).

I also assessed whether or not performance could be improved by incorporating decoding techniques from related ASR systems. Separate from the neural analyses, I trained a phonemic language model (which described the probabilities of observing certain phoneme sequences) on a corpus of over 75 million phonemes. Given a time series of neural activity evoked during perception of a sentence, I implemented Viterbi decoding to predict phoneme sequences using phoneme likelihoods predicted from the neural activity and phoneme transition probabilities from the phonemic language model (Viterbi, 1967; Jurafsky and Martin, 2009). I demonstrated that standard ASR techniques could be used to increase accuracy when decoding phonemes from neural activity.

Additionally, I examined how the gender of the speaker of the acoustic stimuli affected decoding performance. In ASR research, it has been shown that speaker gender can affect the performance of a system designed to decode speech from acoustic signals (Abdulla and Kasabov, 2001). Given the findings that implicate the STG in high-level speech feature processing, it was unclear whether or not speaker gender would have a similar effect when decoding speech from STG activity. After varying which speaker genders were used during model training and testing, I did not find that speaker gender significantly affected phoneme decoding accuracies. This supports previous theories of STG processing, including the ability of the STG to remain invariant to certain speaker-specific acoustic properties, such as pitch (Tang et al., 2017).

This work represented one of the first research efforts in the emerging field of neural speech recognition (NSR), which is a term I use to denote performing speech recognition using neural responses. Other NSR research efforts include the decoding of produced speech from neural activity with a restricted vocabulary (Herff et al., 2015) and the decoding of perceived speech from neural activity in a multi-speaker setting (Chang et al., 2015). A primary goal of NSR research is the development of an advanced speech neuroprosthesis designed to restore the ability to naturally and efficiently communicate to impaired individuals. For example, patients with locked-in syndrome are typically conscious and aware of their surroundings but have little to no voluntary muscle control (American Congress of Rehabilitation Medicine, 1995; Laureys et al., 2005; Bruno et al., 2011; Rousseau et al., 2015; Vansteensel et al., 2016). Assistive speech devices for patients with locked-in syndrome or other forms of paralysis do exist and seem to improve patient quality of life (Bruno et al., 2011; Sellers et al., 2014; Rousseau et al., 2015; Vansteensel et al., 2016), but these devices are slow and unnatural, relying on evoked visual activity and requiring intended messages to be spelled out character-by-character at rates often less than 2 characters per minute. Ideally, patients would have access to a speech prosthesis capable of reliably and efficiently decoding intended speech directly from neural activity with low latency. Although there is likely a multitude of unknown obstacles to overcome before NSR research can produce such a device, the ability to perform real-time, single-trial speech decoding from neural activity is a known requirement. Some research efforts have already demonstrated the ability to process speech-related neural activity in real-time (Leuthardt et al., 2011; Kanas et al., 2014; Cheung and Chang, 2012; Khalighinejad et al., 2017), but no study had shown real-time decoding of words or sentences from neural activity.

In **chapter 3**, I demonstrate classification of aurally presented sentences in real-time from cortical activity (Moses et al., 2018). In this work, I introduced a real-time Neural Speech Recognition (*rtNSR*) software package, which I created as a platform for performing real-time NSR research. I used this package to present multiple repetitions of ten sentences to human subjects while collecting and analyzing ECoG activity. Based on the findings described in chapter 2, I used spatiotemporal feature vectors containing high gamma activity acquired from the STG and neighboring brain regions to fit sentence-level and phoneme-level probabilistic models. During real-time testing with the phoneme-level models, the acquired neural activity was restructured into spatiotemporal feature vectors and used to compute phoneme probabilities at each time point. The time series of predicted phoneme probabilities were then used to compute the probability of each sentence. I demonstrated single-trial sentence classification accuracies of 90% or higher in real-time for both of the subjects that participated in the work (chance accuracy was 10%).

These results show that phoneme representations in the STG could be used to identify which sentence someone heard in real-time. It also validated our *rtNSR* system as a potential platform for future speech prostheses. However, it remains unclear whether or not representations of perceived speech will be useful for future speech prosthetic applications; it seems much more likely that representations of imagined speech will be more relevant for these applications. Some studies have attempted to decode vowels and consonants (Pei et al., 2011) and reconstruct speech spectrograms (Martin et al., 2014) from neural activity evoked during covert speech production (while a subject imagined saying something), and another study has attempted to characterize evoked responses during covert speech perception (while a subject imagined hearing something) and production (Tian and Poeppel, 2010). Although these efforts provided useful insights for imagined speech paradigms, the signal-to-noise ratio of the evoked activity during imagery tasks were typically much lower than the ratios for overt tasks. Whether or not imagined speech can be decoded with existing methods and neural data acquisition technologies remains unknown.

Although additional research into covert speech decoding is warranted, overt speech production remains a viable speech modality for future NSR research. Even though a paralyzed patient might not be able to actually produce speech, investigations of the neural mechanisms of overt speech productions are still relevant because of the volitional, spontaneous nature of overt speech communication. Additionally, the neural correlates of speech articulator control could inform the design of an efficient control algorithm for a speech prosthetic that closely mimics the human articulatory system.

Based on recent findings, the ventral sensorimotor cortex (vSMC) has been shown to have a major role in overt speech production. Enabled by the high spatiotemporal resolution of ECoG signals, one study was able to characterize the spatial and temporal dynamics of phonetic and articulatory representations in the vSMC during speech production (Bouchard et al., 2013). The results of this study suggest that phonetic feature encoding in the vSMC is organized spatially based on articulatory speech features. Later studies expanded on these findings, describing the distributed spatial representations of place of articulation features throughout the sensorimotor cortex using ECoG (Lotte et al., 2015) and functional magnetic resonance imaging (Carey et al., 2017). Other works have demonstrated that high gamma activity recorded from cortical electrodes in the human sensorimotor cortex encode the kinematics of coordinated articulatory movements during production of isolated vowels (Conant et al., 2018) and natural speech (Chartier et al., 2018), including laryngeal movements required for vocal pitch control during production of speech and non-speech vocalizations (Dichter et al., 2018).

Although many of these findings support the theory that neural activity in the sensorimotor cortex encodes articulatory kinematics more strongly than phonetic features (Lotte et al., 2015; Chartier et al., 2018), it has been shown that activity in this region could be used to discriminate between different phonemes during isolated word production (Mugler et al., 2014). When attempting to decode produced speech from brain activity, one advantage of using phonemes instead of articulatory kinematics as a representation of speech is that standard ASR techniques, such as the language modeling and Viterbi decoding techniques described in chapter 2, can included in the decoding approach (well-established techniques to convert articulatory kinematics to text do not currently exist). This phoneme-based approach has been used to decode words from neural activity collected during sentence production (Herff et al., 2015). However, no study has demonstrated the ability to use activity in the vSMC (or other regions) to decode produced words or phrases in real-time.

In **chapter 4**, I demonstrate real-time decoding of perceived and produced speech from cortical activity. After further developing the *rtNSR* system introduced in chapter 3, I used it to collect ECoG activity from the STG, vSMC, and other brain regions while human participants listened to pre-recorded questions and overtly produced answer responses multiple times. Similar to what was done in chapter 2 and chapter 3, I fit phone likelihood models using spatiotemporal neural feature vectors constructed from evoked high gamma activity and time-aligned phonetic transcriptions of the recorded acoustics. Here, phones are similar to phonemes but can indicate additional information about the related speech sound, such as how the vowel was stressed in the word that contained it. During testing, each subject listened to questions and was asked to respond aloud with volitionally chosen answers while their neural activity was collected. High gamma features were extracted from the neural signals in real-time and used to detect when the subject was hearing a question or producing an answer. Each time a speech perception or production event was detected, my *rtNSR* system computed the probability of each question or answer by performing a simplified form of Viterbi decoding using phone likelihoods predicted from the related time segment of high gamma features. Through this approach, I demonstrated reliable decoding of both perceived and produced utterances in real-time.

In addition to separately predicting the question and answer utterances, I was able to use the predicted question likelihoods to inform the answer predictions. Because I designed the task such that certain answers were only reasonable responses to certain questions, I could use the predicted question probabilities to dynamically update the prior probabilities of the answer choices. These answer priors were combined with the predicted answer likelihoods to obtain answer posterior probabilities, which I showed to be more reliable than just the answer likelihoods for each subject. This approach of using the question predictions as context for answer predictions can be relevant for future speech prosthetic applications; it demonstrates that information about the current state of a patient, inferred from neural signals or other sources (such as cameras, microphones, etc.), can be used to improve the decoding of speech from neural activity.

Overall, in this work I focused on using phonetic representations of speech in the high gamma frequency band of evoked cortical activity to decode perceived and produced speech in real-time. I hope that the details provided in the remaining chapters provide a significant contribution to the fields of speech neuroscience, neural engineering, and neural speech recognition. These results are a promising step towards the eventual goal of developing an advanced assistive speech application for many patients suffering from debilitating impairments.

Chapter 2

Neural speech recognition: Continuous phoneme decoding using spatiotemporal representations of human cortical activity

Disclaimer: This chapter is a direct adaptation of the following article:

D. A. Moses, N. Mesgarani, M. K. Leonard, and E. F. Chang, 2016. Neural speech recognition: Continuous phoneme decoding using spatiotemporal representations of human cortical activity. *Journal of Neural Engineering*, 13(5):056004. doi: 10.1088/1741-2560/13/5/056004.

Personal contributions: I developed and tested the decoding system, performed all analyses, and wrote the original draft of the manuscript.

Note: The supplementary material from this article was not included here.

2.1 Abstract

The superior temporal gyrus (STG) and neighboring brain regions play a key role in human language processing. Previous studies have attempted to reconstruct speech information from brain activity in the STG, but few of them incorporate the probabilistic framework and engineering methodology used in modern speech recognition systems. In this work, we describe the initial efforts toward the design of a neural speech recognition (NSR) system that performs continuous phoneme recognition on English stimuli with arbitrary vocabulary sizes using the high gamma band power of local field potentials in the STG and neighboring cortical areas obtained via electrocorticography. The system implements a Viterbi decoder that incorporates phoneme likelihood estimates from a linear discriminant analysis model and transition probabilities from an n-gram phonemic language model. Grid searches were used in an attempt to determine optimal parameterizations of the feature vectors and Viterbi decoder. The performance of the system was significantly improved by using spatiotemporal representations of the neural activity (as opposed to purely spatial representations) and by including language modeling and Viterbi decoding in the NSR system. These results emphasize the importance of modeling the temporal dynamics of neural responses when analyzing their variations with respect to varying stimuli and demonstrate that speech recognition techniques can be successfully leveraged when decoding speech from neural signals. Guided by the results detailed in this work, further development of the NSR system could have applications in the fields of automatic speech recognition and neural prosthetics.

2.2 Introduction

A region of the human auditory cortex called the superior temporal gyrus (STG) is essential for understanding spoken language (Boatman et al., 1997; Binder et al., 2000; Canolty et al., 2007; Rauschecker and Scott, 2009; Pasley et al., 2012; Mesgarani et al., 2014). Previous studies have attempted to reconstruct the acoustics of speech using STG activity (Pasley et al., 2012) and to understand how phonetic features, which are building blocks of spoken language, are encoded in this high-level region of auditory cortex (Mesgarani et al., 2014).

A major focus in the field of automatic speech recognition (ASR) is to develop systems that replicate the human brain's ability to convert acoustic signals into words and sentences. These systems, which have been successfully implemented in multiple industries (BenZeghiba et al., 2007; Kurian, 2014), typically involve the use of probabilistic frameworks and language modeling to decode speech from acoustic signals. Many of the well-established algorithms commonly used in ASR research are reasonably suited for continuous speech decoding tasks using non-acoustic speech-related time series data, such as neural response time series.

A few studies have attempted to use these approaches to decode continuous speech from cortical activity. One group used neural activity recorded during speech production tasks to perform speech decoding with a restricted vocabulary (Herff et al., 2015). Another group focused primarily on decoding speech in a multi-speaker setting using neural activity during speech perception tasks (Chang et al., 2015). Both of these works are examples of an emerging field of study we refer to as neural speech recognition (NSR). We use the term NSR to denote performing continuous speech recognition using neural responses as features. However, to the best of our knowledge, no published work has described the potential benefits of using ASR techniques to decode perceived continuous speech from neural signals in a single-speaker environment. This research direction could add to the field of NSR research by informing the development of a speech decoder that uses neural activity in auditory cortical areas (including the STG) and providing insight on effective representations of neural activity for the purpose of speech decoding.

For these reasons, we developed an initial version of a new NSR system. In its current state, our NSR system uses electrocorticography (ECoG) arrays to decode phoneme sequences from neural populations that respond to perceived speech. Compared to many state-of-the-art ASR systems, which typically incorporate neural network modeling techniques (Hinton et al., 2012; Graves et al., 2013), we designed our NSR system using simple modeling approaches. Relative to the acoustic features typically used in ASR, neural signals that encode speech information are poorly understood, noisy, and available in limited amounts. These factors influenced our decision to use models that are easier to train and interpret and involve fewer tunable parameters. Similarly, our decision to use phoneme-level (as opposed to word-level) decoding in this study, which is a commonly used approach in ASR research, was made for simplicity and in an attempt to gain a better understanding of the limitations of our system. Our primary goal is to help establish an informative foundation for future NSR research by contributing to existing literature in this field. By using optimization techniques to determine effective spatiotemporal feature representations and assessing the impact that individual model components have on the overall performance of the system, we provide novel insights to guide the development of more sophisticated NSR systems.

Future work involving the decoding of speech from neural activity could lead to the development of a speech prosthetic that restores communicative capabilities to impaired individuals, such as those with locked-in syndrome. Locked-in patients are awake and aware of their surroundings but are unable to communicate verbally due to paralysis (Laureys et al., 2005), and only a few methods exist to restore basic communicative functions to locked-in patients (Sellers et al., 2014). These patients could benefit substantially from a device that interprets intended speech based on neural activity and, perhaps through a coupled speech synthesis system, allows more natural communication with others. Although the ideal control paradigm for a successful speech prosthetic is currently unknown, it could rely on covert speech production (Pei et al., 2011; Martin et al., 2014), covert speech perception (Tian and Poeppel, 2010), or an alternative method that has not been described yet. However, because such a device would almost certainly involve processing of neural response time series and probabilistic decoding of speech, we are confident that the approaches and

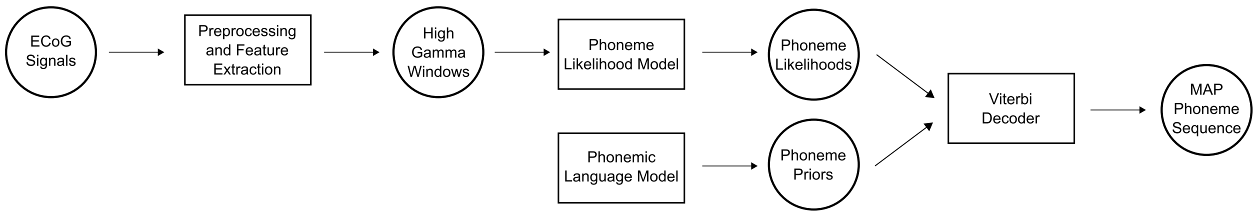


Figure 2.1: A schematic depiction of the NSR system (similar to Figure 9.3 in (Jurafsky and Martin, 2009)). The rectangles signify processing steps and model components, and the circles signify data and computed probability distributions.

Table 2.1: The amount of neural data collected from each subject during perception of the stimuli from the TIMIT and Gump sets. The TIMIT and Gump sets comprised of 499 and 382 unique stimuli, respectively. The table specifies both durations excluding silence samples and total durations in minutes (after rounding). In addition, the total number of stimulus presentations and the mean number of presentations per unique stimulus for each set and each subject are given.

Data set	Subject	Non-silence duration (min)	Total duration (min)	Total stimulus presentations	Mean presentations per stimulus
	A	29	54	1092	2.19
TIMIT	B	31	59	1197	2.40
	C	11	21	412	0.83
	A	47	87	847	2.22
Gump	B	50	92	868	2.27
	C	50	92	867	2.27

results described in this work would be relevant to its design.

An overview of the current NSR system is depicted in Fig. 2.1. First, cortical local field potentials recorded from electrodes over the cortex of multiple subjects (which all include STG coverage) are preprocessed and restructured into high gamma window (HGW) feature vectors, which are spatiotemporal representations of the cortical responses. A phoneme likelihood model, trained using HGWs in conjunction with phonemic class labels, estimates, for each phoneme, the probability of observing an HGW given that it represents a neural response evoked during perception of that phoneme. A separately trained phonemic language model (LM) describes the *a priori* probabilities of different phoneme sequences. Finally, a Viterbi decoder, implementing the well-known hidden Markov model (HMM) architecture, incorporates probabilities from both of these models to yield the maximum *a posteriori* (MAP) phoneme sequence estimate given the input features.

2.3 Materials and methods

2.3.1 Data collection and manipulation

Subjects

The three volunteer subjects (subjects A-C) who participated in this study were human epilepsy patients undergoing treatment at the UCSF Medical Center. ECoG arrays (Ad-Tech, Corp.) were surgically implanted on the cortical surface of each subject for the clinical purpose of localizing seizure foci. Each subject exhibited left hemisphere language dominance, which was determined by clinicians using either the Wada test or fMRI analysis. Prior to surgery, each of these patients gave their informed consent to be a subject for this research. The research protocol was approved by the UCSF Committee on Human Research.

Speech stimuli

For the experimental tasks, each subject listened to multiple speech stimuli. All stimuli were sampled at 16 kHz and presented aurally via loudspeakers at the subject's bedside. Each stimulus contained a speech sample from a single speaker, and the stimuli were separated from each other by at least 500 ms of silence during presentation to each subject. We computed 39-element mel-frequency cepstral coefficient (MFCC) vectors (including energy, velocity, and acceleration features) for each stimulus (Davis and Mermelstein, 1980; Huang et al., 2001; Gold et al., 2011). We used two sets of speech stimuli: the TIMIT set and the Gump set. Information about the number of stimuli presented to and the amount of neural data collected from each subject is given in Table 2.1.

The TIMIT set consisted of phonetically transcribed stimuli from the Texas Instruments / Massachusetts Institute of Technology (TIMIT) database (Garofolo et al., 1993). It contained 499 samples (1.9–3.6 s duration) that had a combined length of approximately 25 minutes and consisted of utterances from 402 different speakers. 354 of the stimuli were each generated by one of the 286 male speakers, and the remaining 145 stimuli were each generated by one of the 116 female speakers. The full stimulus set was not presented to subject C due to external constraints associated with experimentation in a clinical setting (such as clinical interventions and subject fatigue). Most stimuli were presented to each subject multiple times, although the number of presentations of each stimulus varied by subject due to these external constraints. As described in later sections, we used this data set to perform parameter optimization for various components of the NSR system.

The Gump set consisted of re-enacted natural speech samples from Robert Zemeckis's Forrest Gump by two speakers (one male and one female). It contained 91 single word (0.3–1 s duration), 175 phrase (0.4–2.4 s duration), and 116 dialog (4.5–19.9 s duration) speech samples, with each speaker producing 191 of the samples. The combined length across all 382 samples was approximately 43 minutes, with a total of only about 24 minutes when ignoring silence sample points. Each stimulus was presented at least one time to each subject, although the number of presentations of each stimulus varied by subject due to the aforementioned external constraints. We obtained a phonetic transcription for each sample via forced alignment, which was performed using the Penn Phonetics Lab Forced Aligner (Yuan and Liberman, 2008), followed by manual segmentation, which was done in Praat (Boersma, 2001). As explained in Section 2.4, we primarily used this data set to evaluate the performance of the system.

We used a set of 39 phonemic labels in both data sets: 38 phonemes from the Arpabet and /sp/, a "silence phoneme" used to label non-speech data points (Rabiner and Juang, 1993). Some phonetic labels from the TIMIT transcriptions were converted into one of the 39 phonemic labels used in this work. For example, we converted all three of the different silence tokens used in TIMIT transcriptions ("pau", "epi", and "h#") to /sp/. We also

Table 2.2: The phonemes used in this work and their respective categorizations. For visual convenience, the coloring and ordering of the phonemes in this table are used in later figures.

Category	Phoneme
Silence	sp
Stop	b d g p t k
Affricate	ch jh
Fricative	f v s z sh th dh hh
Nasal	m n ng
Approximant	w y l r
Monophthong	iy aa ae eh ah uw ao ih uh er
Diphthong	ey ay ow aw oy

converted each occurrence of /zh/ in the TIMIT set to /sh/ due to its low occurrence rate in the TIMIT set (fewer than 0.15% of time points) and its absence from the Gump set. For analytical purposes, we separated these phonemes into 8 disjoint phonemic categories using descriptive phonetic features (Davenport and Hannahs, 2010). The 39 phonemes and their respective categorizations are shown in Table 2.2.

Neural recordings

Each implanted ECoG array contained 256 disc electrodes with exposure diameters of 1.17 mm arranged in a square lattice formation with a center-to-center electrode spacing of 4 mm. We used these arrays to record cortical local field potentials at multiple cortical sites from each subject during the speech perception tasks. The analog ECoG signals were amplified and quantized using a pre-amplifier (PZ2, Tucker-Davis Technologies), preprocessed using a digital signal processor (RZ2, Tucker-Davis Technologies), and streamed to a separate computer for storage. We acquired and stored the data at a sampling rate of approximately 3052 Hz. Each subject's 3-D pial reconstruction, extracted from T1-weighted MRI data using FreeSurfer (Dale et al., 1999), was co-registered to his or her post-operative computerized tomography scan to determine the ECoG electrode positions on the cortical surface (Hermes et al., 2010). All subjects had unilateral coverage that included the STG; subjects A and B had left hemisphere coverage and subject C had right hemisphere coverage. The reconstruction and electrode positions for each subject appear in Fig. 2.2.

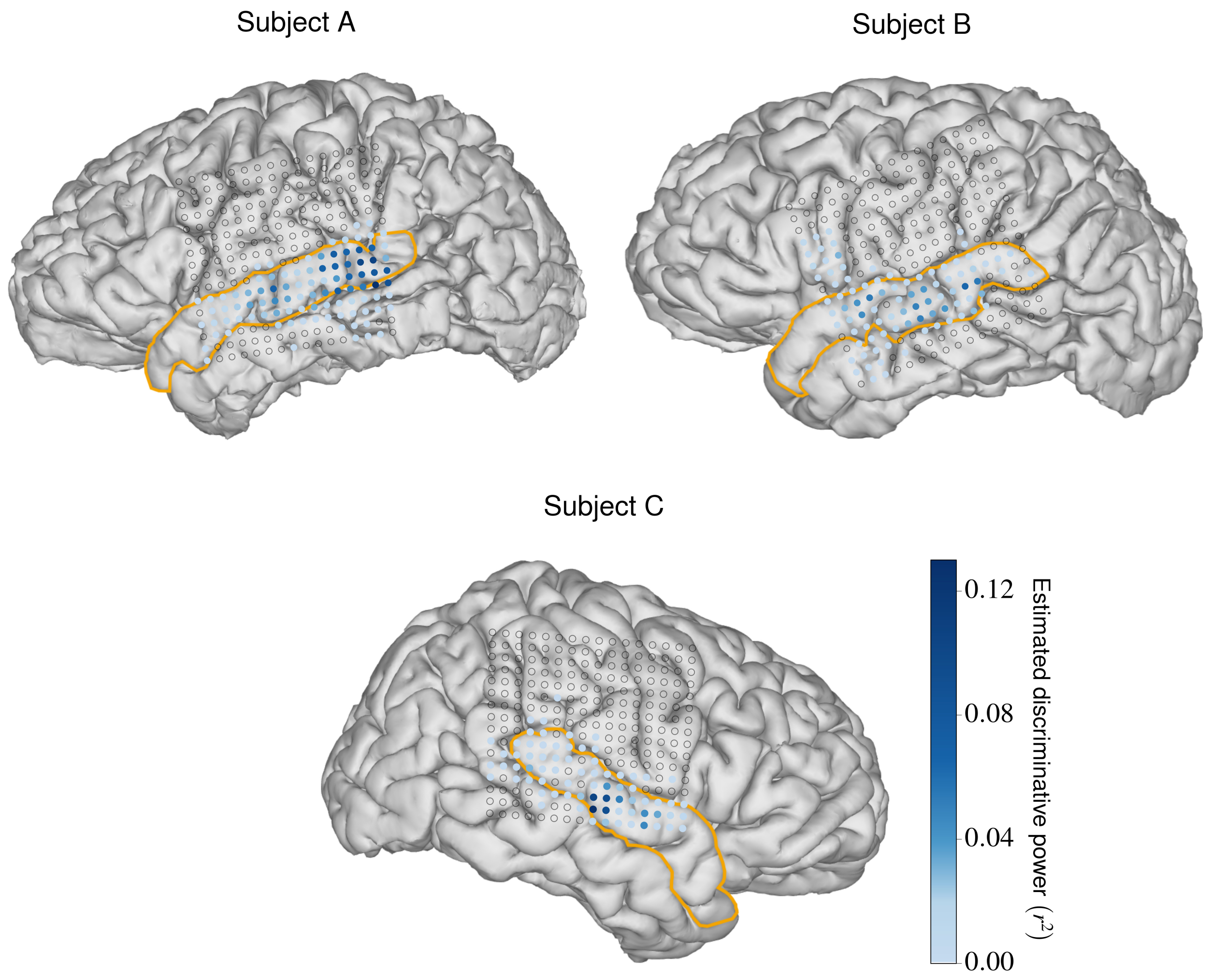


Figure 2.2: MRI reconstructions for each subject with electrode positions superimposed as dots. The sizes of the dots represent the relative sizes of the electrode contacts with respect to the brain. The STG is outlined in orange for each subject. Electrodes that were not deemed relevant appear as circular outlines (electrode relevance is discussed in Section 2.3.1). Relevant electrodes are colored according to their estimated discriminative power (described in Section 2.3.2), depicting the relative importance of each electrode for phoneme discrimination.

Preprocessing

We used MATLAB for preprocessing (The MathWorks Inc., 2013) and Python for all subsequent analyses (unless otherwise specified) (Python Software Foundation, 2010). After data collection, we first down-sampled the raw neural signals to 400 Hz and implemented notch filtering to reduce the mains hum noise at 60 Hz and its harmonics. Next, we qualitatively identified (via visual inspection) channels with severe artifacts and/or significant noise and rejected them. These rejected channels contained time segments that differed greatly in magnitude from the channels that were deemed normal, which is often caused by non-physiological factors (poor electrode contact with the cortical surface, electromagnetic interference from hospital equipment, defective electrodes or wires, etc.). We performed common average referencing on the remaining channels in an attempt to obtain a more favorable spatial representation of the ECoG data (Crone et al., 2001; Ludwig et al., 2009).

Previous research has shown that high gamma band activity (70–150 Hz) correlates strongly with multi-unit firing processes in the brain (Crone et al., 1998) and is an effective representation of brain activity during speech processing (Pasley et al., 2012; Bouchard et al., 2013; Martin et al., 2014; Mesgarani et al., 2014). For these reasons, we applied eight bandpass Gaussian filters with logarithmically increasing center frequencies between 70–150 Hz and semi-logarithmically increasing bandwidths to the neural responses from each electrode channel (Bouchard et al., 2013). These center frequencies, rounded to the nearest decimal place and given in Hz, were 73.0, 79.5, 87.8, 96.9, 107.0, 118.1, 130.4, and 144.0. We then used the Hilbert transform to extract the time-varying analytic amplitudes from each of these eight filtered signals (Marple and Lawrence Marple, 1999; Oppenheim et al., 1999). We down-sampled these eight analytic amplitude signals to 100 Hz and individually z-scored each channel across each experimental session. We used a hyperbolic tangent function to perform soft de-spiking (similar to the methodology used in AFNI's 3dDespike program) on each analytic amplitude signal, which reduced the magnitude of data points more than 10 standard deviations from the mean. We performed singular value decomposition on all eight analytic amplitude signals simultaneously (treating each signal as a single feature) and extracted the first principal component, which we then used to project the eight signals into a one-dimensional space. We used the resulting projection as the representation of high gamma activity in all subsequent analyses.

Since many electrodes for each subject recorded from areas of the cortex that are not associated with speech processing, we decided to only use activity from relevant electrodes during development and testing of the NSR system. Guided by a previously used method to find speech-responsive electrodes (Mesgarani and Chang, 2012), we divided each subject's high gamma activity during perception of the Gump set into speech and silence subsets using the phonemic transcriptions of the stimuli. For each channel, we conducted a t-test that compared all of the samples from each of these two subsets. We considered a channel relevant if the magnitude of the resulting t-value was greater than 2.54, which indicated that the channel was significantly modulated by the presence of a speech stimulus. This threshold value was qualitatively chosen after visual inspection of the high gamma activities for each channel. After these steps, subjects A, B, and C had 95, 89, and 74 relevant channels, respectively. The fewer number of relevant channels for subject C could be a result of electrode coverage over the language non-dominant hemisphere, although existing literature indicates that phonetic processing occurs bilaterally (Hickok and Poeppel, 2007). Over 50% of the relevant channels for each subject were located in the STG. In Fig. 2.2, the relevant electrode locations for each subject are depicted as colored dots and the remaining electrode locations are depicted as circular outlines.

Data reorganization

After preprocessing, a phonemic transcription and associated time sequences of high gamma activity from the relevant electrodes for each subject were available for each acoustic stimulus. An example of one such set of experimental task data is given in Fig. 2.3. This figure includes a visual representation of the phonemic transcription, which is referred to as an "actual posteriogram".

We created 10-fold cross-validation folds for the Gump and TIMIT sets. Each fold contained approximately 90% of the stimuli from the corresponding data set as training

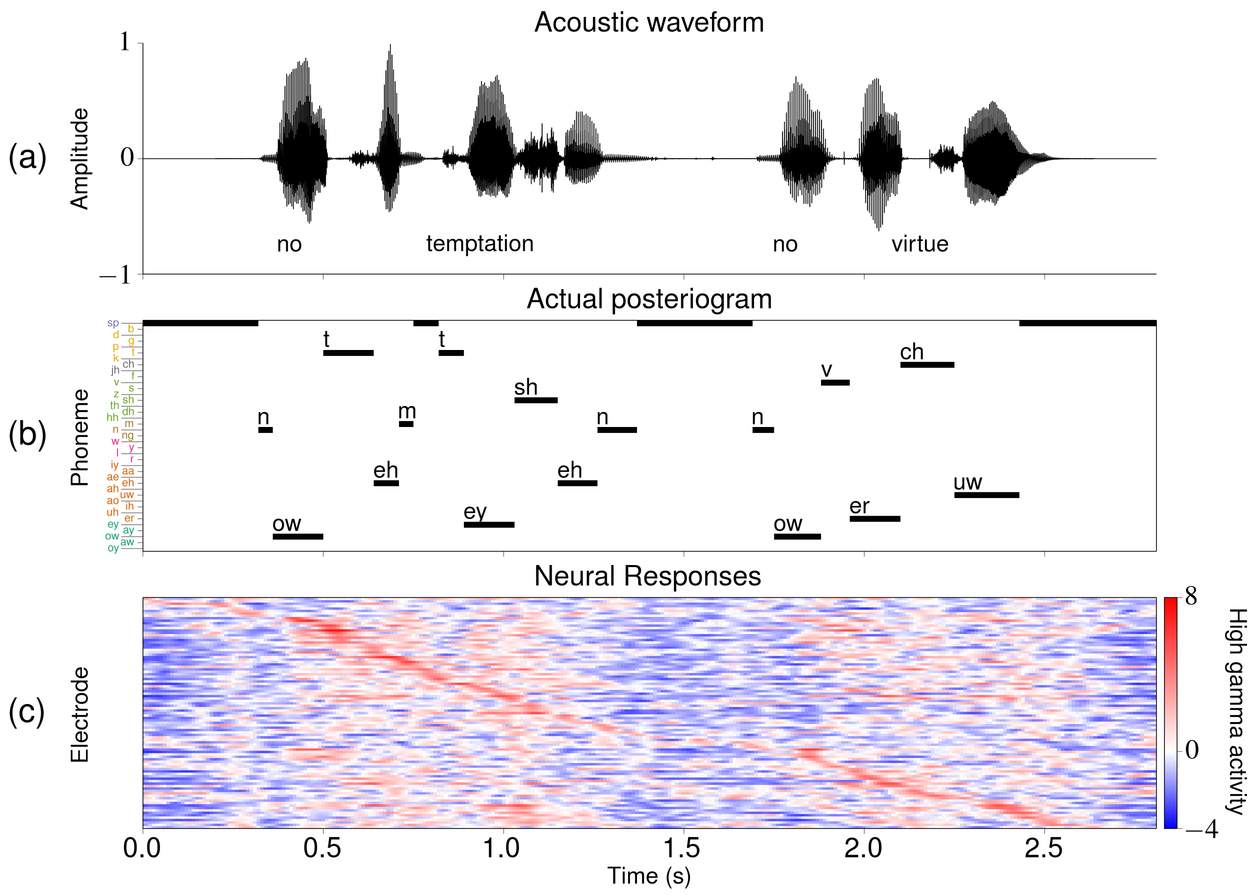


Figure 2.3: Sample task data associated with the utterance "No temptation, no virtue" from the TIMIT set. Some of the silence data points at the start and end of the stimulus were excluded from the visualizations. (*a*) The acoustic waveform along with the associated word transcription. (*b*) The actual posteriogram, which is a visualization of the phonemic transcription associated with this stimulus that depicts which phoneme is specified at each time point in the task. The ordering and coloring of the phoneme labels on the vertical axis are consistent with what was presented in Table 2.2. (*c*) The preprocessed high gamma activity at each of the 95 relevant electrodes for subject A during perception of a single presentation of the stimulus. The electrodes are sorted from top to bottom in ascending peak activity time (i.e. the time at which the electrode exhibited its highest value during this task).

data and the remainder as test data. Each stimulus appeared in the test data for exactly one of the folds. In an attempt to increase homogeneity between folds, we constructed the folds for each of the two data sets such that the numbers of each type of stimuli present in the test data of each fold were approximately equal across all folds. For the 10 TIMIT folds, the two types of stimuli were characterized as being generated by either a male or female speaker. For the 10 Gump folds, the six types of stimuli were characterized as either a word, phrase, or dialog speech sample generated by either the male or female speaker. We performed the majority of our analyses using one or more of these folds.

Feature selection

Auditory speech stimuli evoke complex spatiotemporal cortical responses that can start tens of milliseconds after the acoustic onset and last hundreds of milliseconds after the acoustic offset (Canolty et al., 2007; Engineer et al., 2008; Buonomano and Maass, 2009; Chang et al., 2010; Mesgarani et al., 2014). In an attempt to more accurately model these activation patterns in our NSR system, we used high gamma windows (HGWs) as feature vectors. Each HGW contains multiple data points of high gamma activity within a pre-specified time window across all of the relevant electrodes. Thus, HGWs represent the responses both spatially (by using multiple electrodes) and temporally (by including multiple points in time). Using HGWs as features contradicts a key conditional independence assumption of the HMM architecture utilized by the Viterbi decoder which states that $y_t \perp\!\!\!\perp y_{t-1} \mid q_t$, where q_t and y_t are the phonemic label and feature vector, respectively, at time t. Despite this, we hypothesized that our NSR system would benefit by using these spatiotemporal feature vectors, similar to how performance gains are observed in some ASR systems when velocity and acceleration components are included in the feature vectors (Gold et al., 2011).

The HGWs are parameterized by three values: (1) initial delay, which is the amount of time between the phoneme time point and the first HGW data point, (2) duration, which is the time length of the HGW, and (3) size, which is the number of evenly spaced time points within the time window specified by the first two parameters to include. For example, an HGW parameterized by an initial delay of 50 ms, a duration of 60 ms, and a size of 4 would consist of the data points occurring 50, 70, 90, and 110 ms after the corresponding phoneme time point. We performed grid searches to choose the optimal values for these parameters for each subject. The search included initial delay values between 0–490 ms,

Table 2.3: The results of the feature selection grid searches for each subject. The optimal values for the three HGW parameters found in each grid search are given along with the time offsets calculated from these parameters. The optimal HGS time offset value found in each search is also given.

	High Gamma Window (HGW)				High Gamma Slice (HGS)
Subject	Initial delay (ms)	Duration (ms)	Size (points)	Time offsets (ms)	Time offset (ms)
A	70	180	4	$\{70, 130, 190, 250\}$	100
B	10	230	6	$\{10, 60, 100, 150, 190, 240\}$	90
C	0	210	5	$\{0, 50, 100, 160, 210\}$	120

durations between 0–490 ms, and sizes between 1–25 points. Because the sampling rate of the high gamma activity was 100 Hz, we evaluated the initial delay and duration parameters in 10 ms increments and used rounding when necessary to find indices within the data sequences that most closely corresponded to their related time values. We ignored invalid parameterizations, such as those that used sizes above 4 when the duration was 30 ms. We also ignored parameterizations that included data points occurring 500 ms or more after the corresponding phoneme time point. Any parameterization in which the size was 1 point, which results in a spatial feature vector using the activity at each electrode for a single time point, is referred to as a high gamma slice (HGS). Phoneme likelihood models, which are discussed in Section 2.3.2, were trained and tested with feature vectors constructed using each parameterization. For each subject, we performed a grid search using the neural responses recorded from that subject during each stimulus presentation specified by one arbitrarily-chosen TIMIT cross-validation fold. The performance metric used in each grid search was the average posteriogram accuracy computed from the estimated posteriograms generated for the test data, which is a measure of the frame-by-frame prediction accuracy (see Section 2.4.1 for more details).

The results of these grid searches are given in Table 2.3. The time offsets represent which time points are used when constructing each feature vector. For example, the grid search results for subject A indicate that the optimal HGW for a phoneme occurring at time t contains the high gamma activity values for each relevant electrode at the time points

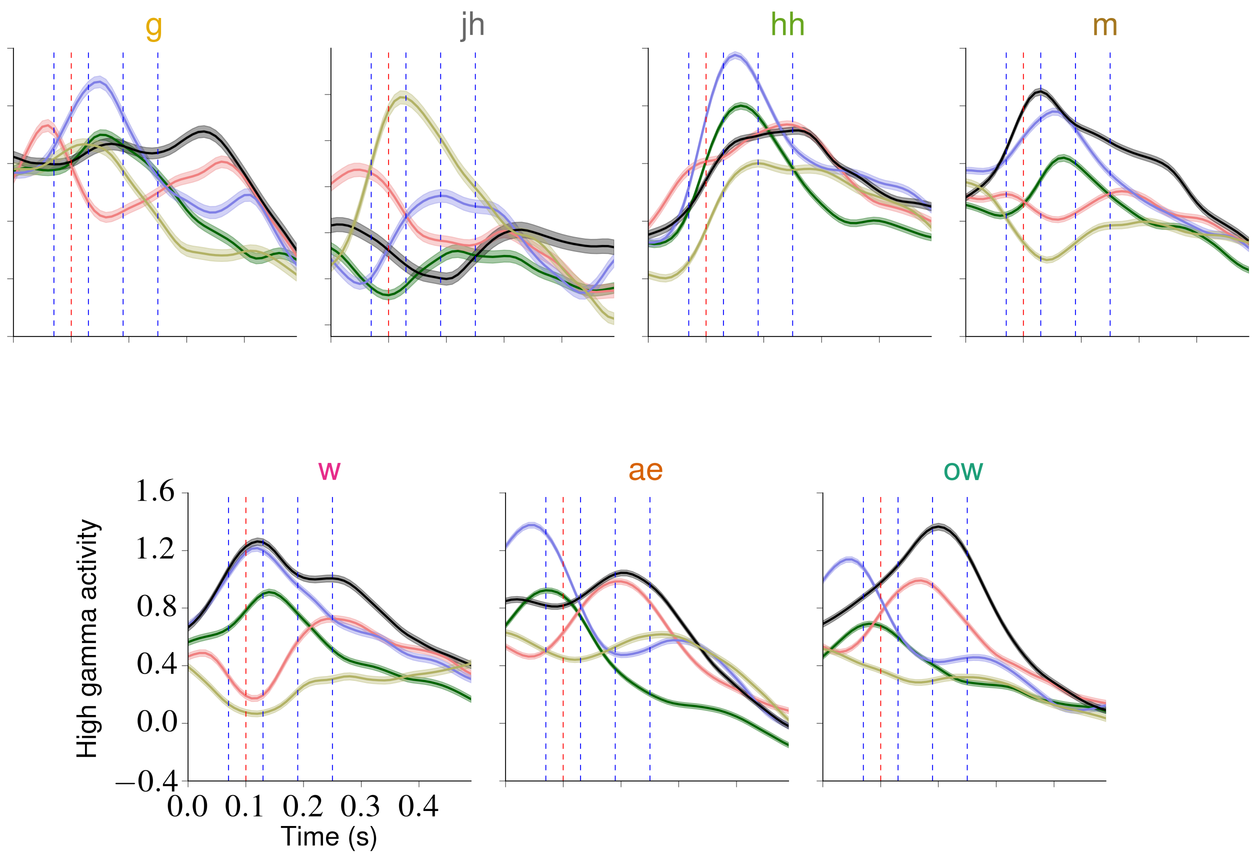


Figure 2.4: A depiction of evoked spatiotemporal response patterns and the computed optimal HGW and HGS parameterizations. At each time point in the TIMIT set specifying one of seven hand-selected phonemes, the high gamma activities recorded from the relevant electrodes for subject A during the succeeding 490 ms were obtained, resulting in approximately 3400 time series per phoneme (on average). From these, the mean response time series for each electrode and phoneme was computed. Each plot contains, for one phoneme, the mean time series for the five electrodes that exhibited the highest estimated discriminative power (as described in Section 2.3.2), depicted as colored curves (the coloring is consistent across the individual plots). One standard error of the mean above and below each electrode curve is included. This subset of seven phonemes contains at least one phoneme from each of the non-silence phonemic categories, and the coloring of the phoneme labels is consistent with the phonemic category coloring introduced in Table 2.2. The optimal HGW contains the values for each electrode at each time point marked with a blue vertical dashed line, and the optimal HGS contains the value for each electrode at the time point marked with a red vertical dashed line at $t = 100$ ms. These plots illustrate the complex spatiotemporal dynamics exhibited by the evoked responses and how these response patterns vary across the phonemes, suggesting that modeling these dynamics (by using HGWs, for example) is beneficial during discrimination tasks.

occurring 70, 130, 190, and 250 milliseconds after t. For later comparison against the system's performance when using the optimal HGW, we also determined the optimal HGS for each subject. For subject A, this occurred at a delay of 100 ms, meaning that the optimal HGS for a phoneme at time t contains the high gamma values for each relevant electrode at $t+100$ ms. Although the optimal HGS time offsets were relatively consistent across subjects, the optimal HGW parameterizations were more varied. This observation could be explained by differences in one or more subject-specific factors, such as electrode coverage, number of relevant electrodes, and cortical structure. The optimal parameterizations for subject A are depicted in Fig. 2.4 along with sample neural response patterns. The results of these grid searches resemble findings reported in related literature (Steinschneider et al., 2011).

2.3.2 NSR system design

Phoneme likelihood model

The phoneme likelihood model used in this NSR system implemented the linear discriminant analysis (LDA) method (Hastie et al., 2009). Although LDA is commonly used as a dimensionality reduction technique, we used it as a classifier trained on the continuous-valued feature vectors (y) and the associated phonemic classes (q). The model fits multivariate Gaussian densities to each class using labeled training data. It assumes that the feature data are normally distributed and that the covariance matrix used to parameterize each class's Gaussian distribution is equal across all classes. An LDA model was chosen due to its simplicity (it has a closed-form solution and no parameters to tune) and its performance in early ASR systems (Haeb-Umbach and Ney, 1992). Additionally, previous work has shown that the STG linearly encodes some phonetic features in the high gamma band (Mesgarani et al., 2014), which helps to motivate the choice of a linear model such as LDA. We implemented this model using the scikit-learn Python package (Pedregosa et al., 2011).

For an unseen feature vector y_t at some time t, the trained LDA model computes likelihood estimates $p\left(y_t | q_t = k\right)$ using the fitted Gaussian density associated with each class (phoneme) k. From this, we can use Bayes' rule to compute the phoneme posteriors

$p\left(q_t = k|y_t\right)$:

$$p\left(q_t = k|y_t\right) \propto p\left(y_t|q_t = k\right) p\left(q_t = k\right), \tag{2.1}$$

where $p\left(q_t = k\right)$ is the prior probability distribution over the phonemic classes. We computed these priors from the relative frequency of each phonemic class in the training data. Note that these priors do not change over time within a single task, but they can change between cross-validation folds. To obtain phoneme posterior probability distributions that sum to one at each time point, we used the following formula:

$$p\left(q_t = k|y_t\right) = \frac{p\left(y_t|q_t = k\right) p\left(q_t = k\right)}{\sum_{l \in Q} p\left(y_t|q_t = l\right) p\left(q_t = l\right)}, \tag{2.2}$$

where Q is the set of all possible phonemic classes.

We also used the LDA model to estimate the discriminative power provided by each electrode channel. For each subject, we trained an LDA model using HGWs and all of the data in the Gump set. Then, for each feature in the LDA model, we computed the variance of the class means, representing a measure of between-class variance for that feature. The values along the diagonal of the shared covariance matrix represented a measure of the within-class variances for each feature (this is only an approximation of within-class variance because we did not force diagonal covariance matrices in the LDA model). The discriminative power for each feature was estimated using the following formula:

$$r_i^2 = 1 - \frac{\sigma_{w,i}^2}{\sigma_{w,i}^2 + \sigma_{b,i}^2}, \tag{2.3}$$

where r_i^2, $\sigma_{w,i}^2$, and $\sigma_{b,i}^2$ are the estimated discriminative power, within-class variance, and between-class variance, respectively, for the ith feature. For each electrode, the r^2 values for each feature that specified a time point for that electrode in the HGW were averaged, yielding an estimated discriminative power for that electrode. For each subject, the relevant electrodes in the STG accounted for more than two-thirds of the total estimated discriminative power across all relevant electrodes. The r^2 values for each relevant electrode for each subject are depicted in Fig. 2.2.

Phonemic language model

The language model (LM) used in the NSR system provides estimates for the *a priori* probabilities of phonemic sequences. Phoneme LMs are typically trained on large corpora containing phoneme sequences. We decided to construct a phoneme corpus by phonemically transcribing English sentences contained in the SUBTLEX-US corpus, which was created using the subtitles from many American films and television series (Brysbaert and New, 2009). The Festival speech synthesis system was used to convert the sentences into phoneme sequences (Black et al., 1997). Some sentences were excluded, such as short sentences with fewer than 6 phonemes and sentences that the Festival system was not able to phonetize. All /ax/ and /zh/ phoneme tokens were converted to /ah/ and /sh/ tokens, respectively. Any phoneme sequence which exactly matched the phoneme sequence associated with any of the Gump stimuli was excluded in an effort to keep the LM more generalized. A silence phoneme (/sp/) token was inserted at the end of the phoneme sequence transcribed from each sentence so that the sequences could be combined into one large corpus. Approximately 4.3 million sentences were included, resulting in a phoneme corpus with about 76.9 million non-silence phonemes (a total of about 81.2 million phoneme tokens when /sp/ is included).

Because of the relatively simple implementation and robust performance of n-gram LMs, we decided to choose between one of two different types of interpolated n-gram LMs for use in the NSR system: a basic n-gram LM using additive smoothing (Lidstone, 1920; Chen and Goodman, 1998) and a modified Kneser-Ney n-gram LM (Kneser and Ney, 1995; Chen and Goodman, 1998). Although the modified Kneser-Ney n-gram LM typically outperforms other n-gram LMs when used in word-level ASR systems, it might not be as suitable for

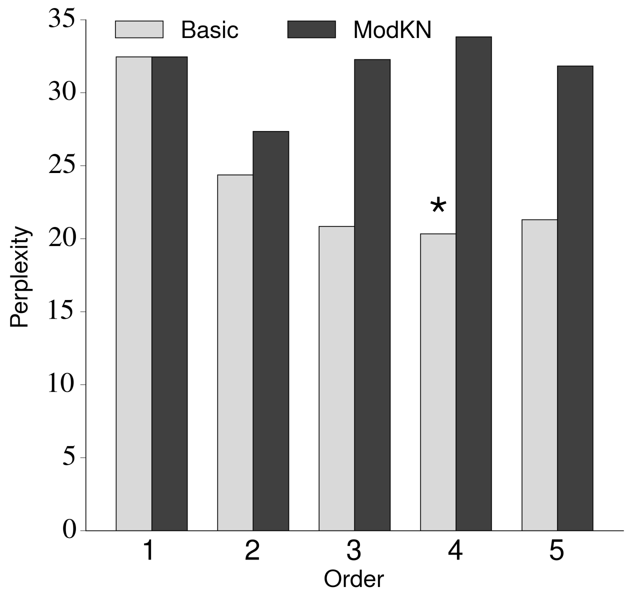


Figure 2.5: A comparison of the basic and modified Kneser-Ney n-gram phonemic language models using multiple orders. The labels “Basic” and “ModKN” refer to the basic additive smoothing and modified Kneser-Ney n-gram LMs, respectively. The same training and testing corpora were used for each LM. A lower perplexity value indicates better performance. The basic additive smoothing 4-gram model (marked with an asterisk) exhibited the best performance, with a perplexity of 20.33.

phoneme-level decoding because of the relatively small number of tokens (the 39 phoneme tokens) we use in our NSR system. We compared the performance of these two types of LMs using orders of $n \in \{1, 2, 3, 4, 5\}$. Each LM was trained using the aforementioned corpus and tested on a phoneme corpus constructed by concatenating the phonemic transcriptions of the 499 stimuli in the TIMIT set (including a silence phoneme between stimuli). We used the perplexity of the LM on the test corpus as the evaluation metric (a lower perplexity indicates better performance) (Chen and Goodman, 1998). Given the results of this analysis, which are depicted in Fig. 2.5, we decided to use the basic 4-gram LM (trained on the previously described phoneme corpus) in our NSR system.

For a given sequence of phonemes, the basic 4-gram LM provides conditional probability estimates of $p\left(q_i = k | q_{i-3}^{i-1}\right)$ for each phonemic class k at each index i within the sequence, where the notation q_a^b denotes the ath through the bth phonemes in the sequence. The phonemic sequences used in LMs contain no information about phoneme durations; a phoneme that spans any number of time points will be represented as a phoneme at a single index in these phonemic sequences. For notational simplicity, these conditional probabilities are sometimes represented as $p\left(q_t\right)$, which suppresses their implicit dependence on the three distinct phonemes that precede the phoneme at time t. These probabilities should not be confused with the priors discussed in the previous section.

In general, the conditional probabilities for a basic additive smoothing n-gram LM are computed recursively using the following formula (Chen and Goodman, 1998):

$$p\left(q_i = k | q_{i-n+1}^{i-1}\right) = \begin{cases} \lambda_n \left[\frac{\delta + c\left(q_{i-n+1}^{i}\right)}{\delta|Q| + c\left(q_{i-n+1}^{i-1}\right)}\right] + \left(1-\lambda_n\right) p\left(q_i = k | q_{i-n+2}^{i-1}\right) & \text{for } n > 1 \\ \frac{\delta + c\left(q_i\right)}{\delta|Q|} & \text{for } n = 1. \end{cases} \tag{2.4}$$

Here, $c\left(q_a^b\right)$ is the count of the number of times the n-gram q_a^b occurs in the corpus, δ is the additive smoothing factor that is added to the count of each n-gram (typically $0 \leq \delta \leq 1$), Q is the set of all possible phonemes, and λ_n is the interpolation weight for the order n. In our NSR system, we used $n = 4$, $\delta = 0.1$, and $\left[\lambda_4, \lambda_3, \lambda_2\right] = \left[\frac{5}{9}, \frac{4}{7}, \frac{3}{5}\right]$.

Viterbi decoder

We implemented a Viterbi decoding algorithm to provide MAP phoneme sequence estimates given a sequence of likelihood estimates (from the likelihood model) and phoneme transition probabilities (from the LM) (Viterbi, 1967; Jurafsky and Martin, 2009). The algorithm uses Viterbi path probabilities, which are computed recursively using the following formula:

$$v_t\left(j\right) = v_{t-1}\left(i\right) + \log p\left(y_t | q_t\right) + L \log p\left(q_t\right) + Pn_t. \tag{2.5}$$

Here, $v_t\left(j\right)$ is the jth Viterbi path's log probability at time t, $v_{t-1}\left(i\right)$ is the ith Viterbi path's log probability at time $t-1$, $p\left(y_t | q_t\right)$ is the likelihood of observing the feature vector y_t given

q_t (provided by the phoneme likelihood model), $p\left(q_t\right)$ is the prior probability of observing phoneme q_t at time t (provided by the LM), L is the language model scaling factor (LMSF), P is the phoneme insertion penalty, and n_t is an indicator variable that is 1 if and only if q_t for path j is not equal to q_{t-1} for path i. At every time point, the likelihoods $p\left(y_t|q_t\right)$ are normalized such that they sum to one across all phonemes.

Paths are computed for each combination of $i \in \{1, 2, \ldots, I_{t-1}\}$ and $q_t \in Q$, where I_{t-1} is the total number of paths at time $t-1$ and Q is the set of all possible phonemic classes. This results in a new path for each $j \in \{1, 2, \ldots, |Q|\, I_{t-1}\}$ at time t. For example, if there are 12 paths at time $t-1$ and 39 phonemes, then there will be 468 paths at time t (prior to pruning). We used log probabilities for computational efficiency and numerical stability. To initialize the recursion, we forced each decoding to start at time $t = 0$ with $q_0 = /\mathrm{sp}/$ and a possible Viterbi path set of $\{v_0\left(1\right) = 0\}$. After computation of all of the $v_t\left(j\right)$ log probabilities for each t, we performed two steps of pruning. First, we performed a beam search to prune unlikely paths between iterations by discarding paths that did not satisfy

$$v_t\left(j\right) \geq \left[\max_z v_t\left(z\right)\right] - c. \tag{2.6}$$

Here, z indexes over all paths available at time t and c is the beam search criterion, which we set equal to 50. Afterwards, we only retained a maximum of 100 of the most likely paths between iterations. The decoded MAP phoneme sequence is specified by the path at index m at time T, where $m = \operatorname*{argmax}_u v_T\left(u\right)$, T is the final time point, and u indexes over all paths available at time T.

The three main tunable parameters of this Viterbi decoding algorithm are the LMSF, the phoneme insertion penalty, and the self-transition probabilities. The LMSF controls the relative strength of the LM (as compared to the strength of the phoneme likelihood model). Because the normalized likelihoods $p\left(y_t|q_t\right)$ and LM probabilities $p\left(q_t\right)$ each sum to one at every time point, the LMSF represents the ratio of the strength of the LM to the strength of the likelihood model. The phoneme insertion penalty (P) controls the preference for decoding short vs. long phoneme sequences. The self-transition probability (s) specifies the probability at each time point that a self-transition will occur, which is important because phonemes typically last for more than one time point. This probability replaces the probabilities given by the LM for $p(q_t)$ when $q_t = q_{t-1}$. Note that in the context of phoneme-level decoding (as opposed to word-level decoding), the self-transition probability is similar to the phoneme insertion penalty in that it also controls the preference for decoding short vs. long phoneme sequences.

We used grid searches to determine the optimal values for these three parameters. For each subject, we obtained likelihood estimates at each time point for each TIMIT stimulus presentation. These likelihoods were obtained from likelihood models trained with HGW feature vectors using the TIMIT cross-validation scheme. We also obtained likelihood estimates using feature vectors from all subjects simultaneously (as described in Section 2.4) and using MFCC features. Using these likelihood estimates, we evaluated the performance of the decoder when parameterized by all possible combinations of $L \in \left\{0, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1, 2, 3, 4, 5\right\}$, $P \in \{-7, -6, \ldots, 0, 1, 2\}$, and $s \in \{0, 0.1, \ldots, 0.8, 0.9\}$. The performance metric used was the value of the expression $(1 - \epsilon) + \gamma$, where ϵ and γ are the average phoneme error rate and posteriogram accuracy, respectively, across all of the results for a given parameterization (these metrics are explained in Section 2.4.1). The results of this grid search for each subject, for all subjects simultaneously, and for the acoustic features are given in Table 2.4.

Table 2.4: The optimal values for the three Viterbi parameters found by the grid searches using MFCC features, neural features for each subject, and neural features for all subjects simultaneously.

Subject	LMSF (L)	Phoneme insertion penalty (P)	Self-transition probability (s)
MFCC	2	-1	0.4
A	2	-1	0.3
B	2	-1	0.9
C	3	-1	0.1
All	2	-2	0.4

2.4 Results

We evaluated the performance of the NSR system using multiple feature sets and metrics. Each evaluation used all 10 of the Gump cross-validation folds. We conducted evaluations using either HGSs or HGWs as feature vectors. For each subject, we performed evaluations with single-trial data (using high gamma activity from each stimulus presentation individually) and averaged data (using high gamma activity averaged across all of the presentations of each stimulus). In addition to these analyses using responses from individual subjects, we also performed evaluations using concatenated feature vectors from all of the subjects; because the neural response data are time-aligned to the stimuli and the stimuli are identical across subjects, we were able to generate feature vector time sequences using data from all of the subjects simultaneously for each stimulus by concatenating the feature vectors (averaged across stimulus presentations) from each subject during perception of the stimulus.

For each evaluation, we obtained results using two types of predictions: “estimations” and “decodings”. Here, estimations refer to the phoneme sequences constructed by choosing the most likely phoneme at each time point from the phoneme posterior probabilities provided by the LDA model, and decodings refer to the MAP phoneme sequences provided by the Viterbi decoder. Continuing with the notation introduced in Section 2.3.2, we computed the estimated phoneme sequences using $\hat{q}_t = \underset{k}{\operatorname{argmax}}\, p\left(q_t = k | y_t\right)$ at each time point, where $\hat{q}_t$ is the estimated phoneme at time t in one of the stimuli. By comparing the estimation and decoding results, it is possible to measure the impact that language modeling and Viterbi decoding had on the performance of the system.

We performed a separate evaluation that used MFCCs as features to assess how well a similarly-designed ASR system would perform on the stimuli. Additionally, we evaluated chance performance by decoding the non-silence phoneme with the most time points in the training set, which was always /s/, at each time point. When considering frame-by-frame accuracy, this is a more conservative method of chance performance than simply using 1 divided by the number of classes (which would be about 2.6% for the 38 non-silence phonemes). We assessed the performance of the NSR system using three evaluation metrics to measure the similarities between the predicted and actual phoneme sequences: phoneme error rate, posteriogram accuracy, and confusion accuracy.

2.4.1 Evaluation metrics

Phoneme error rate

In ASR research, the word error rate evaluation metric is commonly used to assess the performance of a speech recognition system (BenZeghiba et al., 2007). One of the main advantages of using this metric is that it evaluates performance by directly using predicted word sequences, which are what end users of many ASR systems interact with. The analog of this metric when used in the context of phoneme-level recognition is the phoneme error rate (PER), which is a measure of the Levenshtein distance between actual and predicted phoneme sequences. The PER for a predicted phoneme sequence can be computed using the following formula:

$$\text{PER} = \frac{S + D + I}{N} \tag{2.7}$$

Here, S, D, and I specify the minimum number of substitutions, deletions, and insertions (respectively) required to transform the predicted phoneme sequence into the reference (actual) sequence, and N denotes the number of phonemes in the reference sequence. A lower PER value signifies better performance. Note that it is possible for PER values to exceed 1.0; for example, if the predicted sequence was /ay n ow/ and the reference sequence was /ay/, the PER value would be 2.0, with $S = I = 0$, $D = 2$, and $N = 1$.

The PER metric uses sequences that have been "compressed" by removing all silence phonemes from the sequences and then traversing each sequence in order and removing any phoneme that occurs immediately after an identical phoneme. Therefore, compressed sequences do not contain information about the time durations of any item in the sequence, which is typically not relevant for the end users of ASR systems. Note that the PERs for estimation results tend to be relatively large and are primarily included in the results for completeness; typically, PERs are only informative for decoding results.

Posteriogram accuracy

We constructed estimated and decoded posteriograms, which use estimated and decoded phoneme sequences, respectively, to visually represent predicted phoneme sequences. Sample posteriograms, along with the related time sequence of phoneme posteriors, are given in Fig. 2.6. The posteriogram accuracy is a measure of the frame-by-frame accuracy of a predicted posteriogram; it represents the fraction of time points within a given stimulus for which the predicted phoneme was equal to the actual phoneme. This metric does not use compressed sequences and is sensitive to the time durations of the predicted phonemes. For this metric, we excluded any data points for which the actual phoneme was the silence phoneme; although detecting the absence of speech will likely be an important aspect of an applied NSR system, including these points led to performance overestimation due to increased posteriogram accuracy for each prediction.

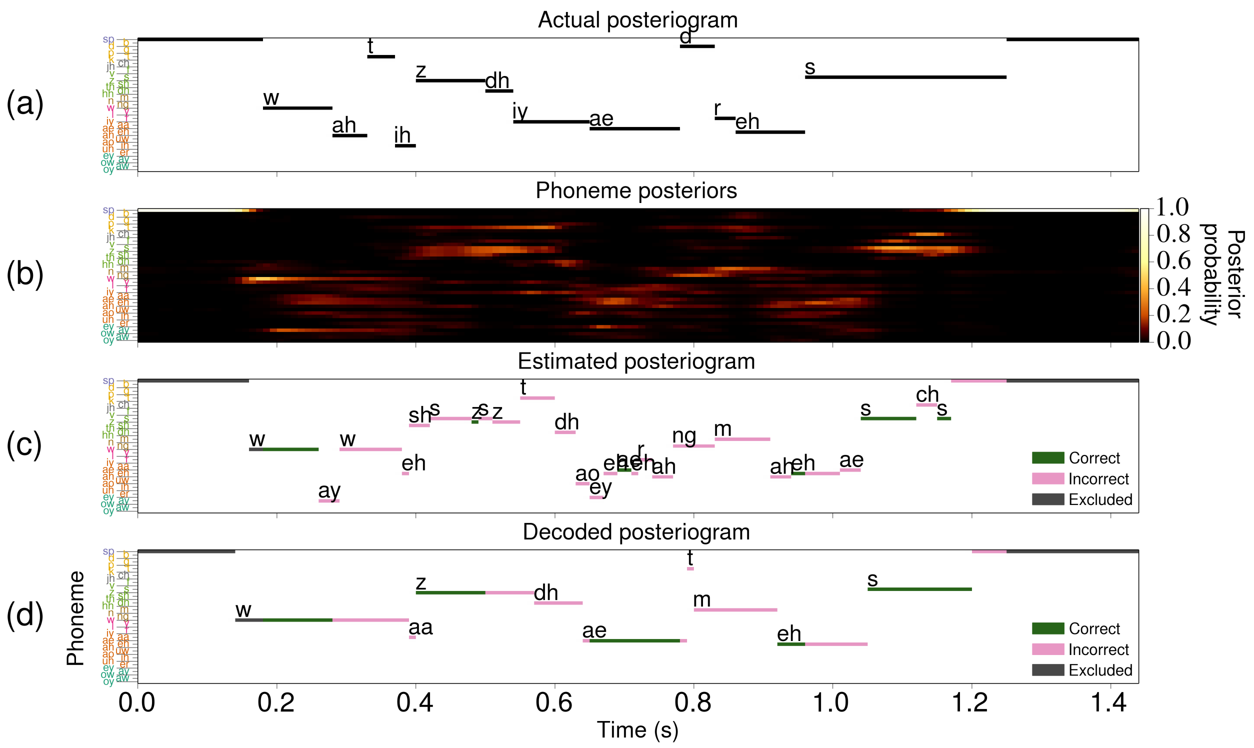


Figure 2.6: A sample set of results obtained using HGWs from subject A during perception of the utterance "What is the address?" by a female speaker in the Gump set. In all four visualizations, the ordering and coloring of the phoneme labels given in Table 2.2 are used. Some of the data points specifying silence at the start and end of the stimulus were excluded from the visualizations. (*a*) The actual posteriogram showing the phonemic transcription of the stimulus. (*b*) The phoneme posterior probability distribution at each time point during the task. (*c*) The estimated posteriogram constructed by classifying the phoneme posteriors in (*b*) using the most likely phoneme at each time point. Here, the green and pink points signify classifications that were considered correct and incorrect, respectively. Dark gray points signify data that were excluded from the calculation of the posteriogram accuracy. (*d*) The decoded posteriogram computed by the Viterbi decoder, which is represented using the same coloring scheme as the estimated posteriogram. For this specific decoding result, the posteriogram accuracy is 48.6% and the phoneme error rate is 50.0%.

Confusion accuracy

We computed phoneme confusion matrices, such as the ones shown in Fig. 2.7, for each evaluation using the confusions between the actual and predicted phonemes for each time point in each stimulus. We normalized the confusion matrices by row such that the confusion values for any actual phoneme would sum to 1 across all of the predicted phonemes. The values along the diagonal of the matrix are measures of the model's ability to correctly classify each phoneme. The confusion accuracy is defined as the mean of these values,

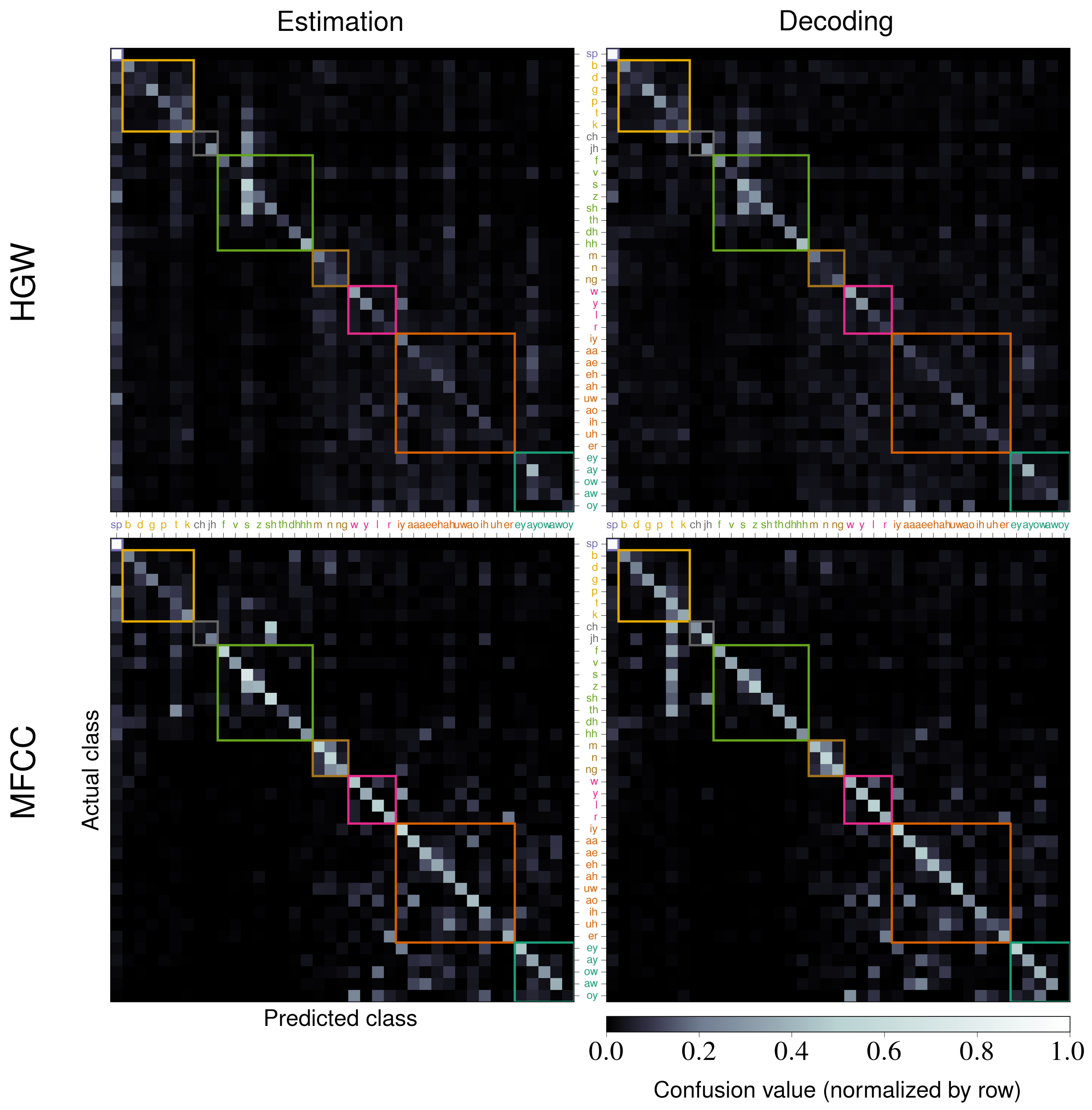


Figure 2.7: A sample set of confusion matrices computed using the performance evaluation results. The top row contains results using averaged HGWs from subject A and the bottom row contains results using MFCCs. The left column contains estimation results and the right column contains decoding results. The color-value mapping is identical across all matrices and uses row-normalized confusion values. The colored square outlines signify phonemic categories and correspond to the ordering and coloring of the phonemes on both axes (which match what was given in Table 2.2). Confusion accuracies were computed by taking the mean value along the diagonal (excluding the silence phoneme value).

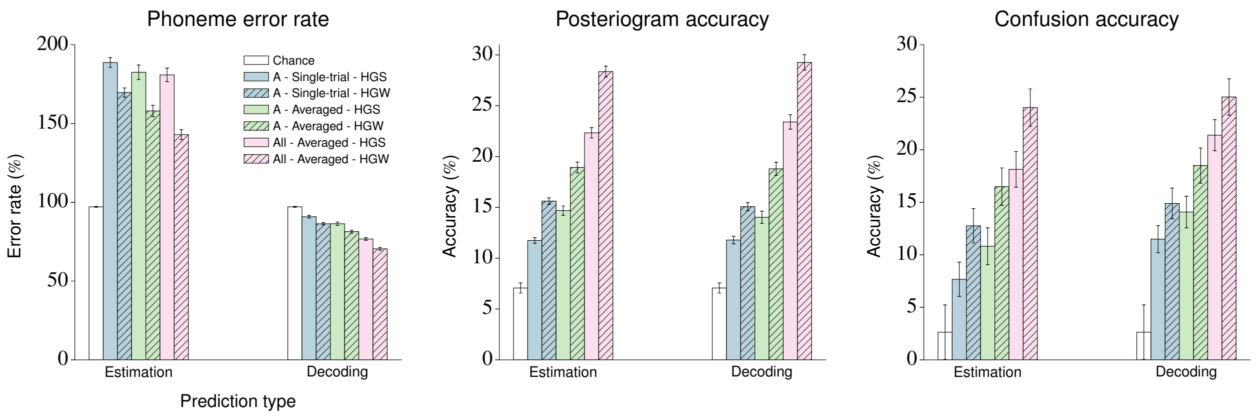


Figure 2.8: Visualization of the performance evaluation of the NSR system on the stimuli within the Gump set using single-trial and averaged HGSs and HGWs from subject A and concatenated feature vectors across all subjects. Chance performance is also included. Error bars indicate standard error of the mean. The results for all of the subjects with standard deviations are given in Table 2.5.

which is effectively a re-scaled measure of the matrix's trace. Consequently, this metric does not directly depend on the number of available time points for each phoneme; it weighs the classification accuracy for each phoneme equally. This metric can be used to identify whether or not the system only successfully predicts common phonemes, which would negatively affect the confusion accuracy more drastically than posteriogram accuracy. We also excluded the value along the diagonal for the silence phoneme when computing this metric to prevent performance overestimation.

2.4.2 System performance

The performance of the NSR system for subject A and for the concatenated feature vectors is depicted in Fig. 2.8. The results of the system's full performance evaluation are summarized in Table 2.5.

In the figure and the table, the statistics for the phoneme error rate and posteriogram accuracy metrics are computed using the individual results from each stimulus, and the statistics for the confusion accuracy metric are computed using the values along the diagonal

Table 2.5: Performance evaluation of the NSR system on the stimuli within the Gump set using three different types of feature vectors (MFCCs, HGSs, and HGWs) across four different subject sets (the three individual subjects and one combination of all subjects). Both single-trial and averaged neural response feature vectors were evaluated. Chance performance, which involves predicting the most likely phoneme /s/ at each time point, is also included. The phoneme error rate, posteriogram accuracy, and confusion accuracy metrics are used to assess the estimation and decoding results. All results are given as percentages in the following form: mean ± standard deviation.

Feature set		Results					
		Estimation			Decoding		
Feature type	Subject (or all)	Phoneme error rate (%)	Posteriogram accuracy (%)	Confusion accuracy (%)	Phoneme error rate (%)	Posteriogram accuracy (%)	Confusion accuracy (%)
Chance	-	97.12 ± 5.34	7.06 ± 9.77	2.63 ± 16.01	97.12 ± 5.34	7.06 ± 9.77	2.63 ± 16.01
MFCC	-	208.16 ± 67.45	40.45 ± 10.76	34.51 ± 16.15	60.60 ± 25.16	41.88 ± 17.30	36.30 ± 10.86
Single-trial HGS	A	188.73 ± 90.28	11.74 ± 7.98	7.67 ± 10.03	90.84 ± 25.00	11.79 ± 10.88	11.49 ± 7.94
	B	176.91 ± 114.31	7.72 ± 6.02	4.79 ± 4.96	92.31 ± 26.43	10.23 ± 11.06	9.63 ± 5.52
	C	173.83 ± 92.48	9.00 ± 6.76	4.80 ± 7.22	87.67 ± 17.36	10.70 ± 11.58	8.16 ± 6.23
Single-trial HGW	A	169.68 ± 84.89	15.62 ± 9.21	12.76 ± 10.12	86.34 ± 22.07	15.06 ± 11.73	14.89 ± 9.01
	B	151.70 ± 78.01	12.69 ± 8.18	9.58 ± 5.89	90.63 ± 26.28	12.71 ± 11.49	11.80 ± 5.96
	C	152.48 ± 75.98	12.93 ± 8.77	8.09 ± 8.22	85.71 ± 17.30	12.62 ± 12.10	9.84 ± 6.40
Averaged HGS	A	182.53 ± 89.81	14.68 ± 8.87	10.81 ± 10.80	86.37 ± 20.83	14.03 ± 11.72	14.07 ± 9.28
	B	182.80 ± 99.75	11.95 ± 7.82	8.21 ± 6.92	87.13 ± 18.01	13.51 ± 11.30	12.59 ± 7.26
	C	181.75 ± 78.74	12.22 ± 8.01	7.08 ± 9.01	85.36 ± 13.67	12.41 ± 11.45	10.01 ± 6.56
	All	180.87 ± 84.47	22.34 ± 9.94	18.14 ± 10.47	76.68 ± 15.82	23.41 ± 13.85	21.39 ± 9.12
Averaged HGW	A	158.00 ± 69.86	18.93 ± 10.39	16.49 ± 10.99	81.39 ± 18.85	18.79 ± 12.57	18.49 ± 10.30
	B	149.66 ± 65.27	17.35 ± 9.46	13.80 ± 7.30	82.84 ± 21.02	17.64 ± 13.31	15.30 ± 6.99
	C	157.35 ± 73.80	16.42 ± 9.84	11.20 ± 8.79	83.65 ± 19.18	15.36 ± 13.02	12.27 ± 7.99
	All	142.96 ± 61.31	28.36 ± 10.47	24.02 ± 10.99	70.47 ± 16.77	29.26 ± 14.88	25.03 ± 10.69

of the overall confusion matrix.

We performed a variety of statistical significance tests on these results, using the one-tailed Wilcoxon signed-rank test (abbreviated to Wilcoxon) for paired comparisons and the one-tailed Welch's t-test (abbreviated to Welch's) for unpaired comparisons. We use a significance level of $\alpha = 0.01$ during assessment of our results.

All of the HGW decoding results were significantly better than the HGS estimation results for all subject sets (each individual subject and the concatenated features) and metrics (Wilcoxon, $p < 10^{-6}$).

All of the HGW results were significantly better than the HGS results for all subject sets and metrics (Wilcoxon, $p < 0.005$).

The decoding results were significantly better than the estimation results when evaluated with the PER metric for all subject sets (Wilcoxon, $p < 10^{-11}$). Similarly, confusion accuracies were significantly better for decoding results than for estimation results (Wilcoxon, $p < 0.01$) for all evaluations except the ones using averaged HGWs from subject C and concatenated HGWs. However, significant improvements were not observed for many of the evaluations when using the posteriogram accuracy metric, and in some instances the mean decoding posteriogram accuracies were lower than the estimation posteriogram accuracies.

Averaged neural feature vectors outperformed their single-trial counterparts for each subject when using the posteriogram accuracy metric (Welch's, $p < 0.01$). Except when HGWs from subject C are used, this was also observed for comparisons involving decoding PERs (Welch's, $p < 0.01$). This was not observed for the majority of the confusion accuracy or estimation PER comparisons.

The concatenated feature vectors performed significantly better than individual subject feature vectors when evaluated with each metric other than the estimation PER metric (Wilcoxon for averaged results, Welch's for unaveraged results, $p < 10^{-5}$).

MFCC features significantly outperformed neural features when evaluated with each metric other than the estimation PER metric (Wilcoxon for averaged results, Welch's for unaveraged results, $p < 10^{-5}$).

Neural features performed better than chance in most situations (Wilcoxon for averaged results, Welch's for unaveraged results, $p < 0.01$). The exceptions comprised of all estimation PER comparisons, a subset of the single-trial confusion accuracy results, and the single-trial estimation posteriogram accuracy result with HGSs for subject B.

2.4.3 Phoneme time position effects

Previous research has shown that transient responses to the onset of an acoustic stimulus are exhibited by some neurons in the auditory cortex of rats (DeWeese et al., 2003; Ogawa et al., 2011) and humans (Tiitinen et al., 2012; Okamoto and Kakigi, 2014). If similar response patterns are present in our ECoG data, we can expect the performance of our phoneme likelihood estimator to vary throughout the duration any given utterance. Specifically, we hypothesize that NSR performance degrades over the course of an utterance due to temporal complexities present in the evoked neural response patterns, such as sensitivity to stimulus onsets. Additionally, we expect that these same effects are not present in the acoustic features.

To assess this hypothesis, we analyzed the impact that phoneme time position had on posteriogram accuracy for both acoustic and neural features. Here, the time position of a phoneme is equal to the amount of elapsed time between the utterance onset and the phoneme time point. We defined the onset of an utterance to be the onset of any non-silence phoneme that occurs immediately after a period of silence lasting 500 ms or longer (we did not simply use the first non-silence phoneme in each stimulus because some of the longer Gump stimuli contained multiple sentences). For each feature type, we used the estimated posteriograms generated for the 382 Gump stimuli to constructed a data set using each time point in the corresponding actual posteriograms specifying a non-silence phoneme. Each datum in this new data set specified the stimulus identity, the phoneme time position, and a binary indicator that was 1 if that time point was correctly classified in the estimated posteriogram and 0 otherwise. A depiction of these data sets for MFCCs and averaged HGWs from subject B is provided in Fig. 2.9.

For each feature type, we used the lme4 package within the R programming language (R Core Team, 2015; Bates et al., 2015) to fit a mixed effects logistic regression model with

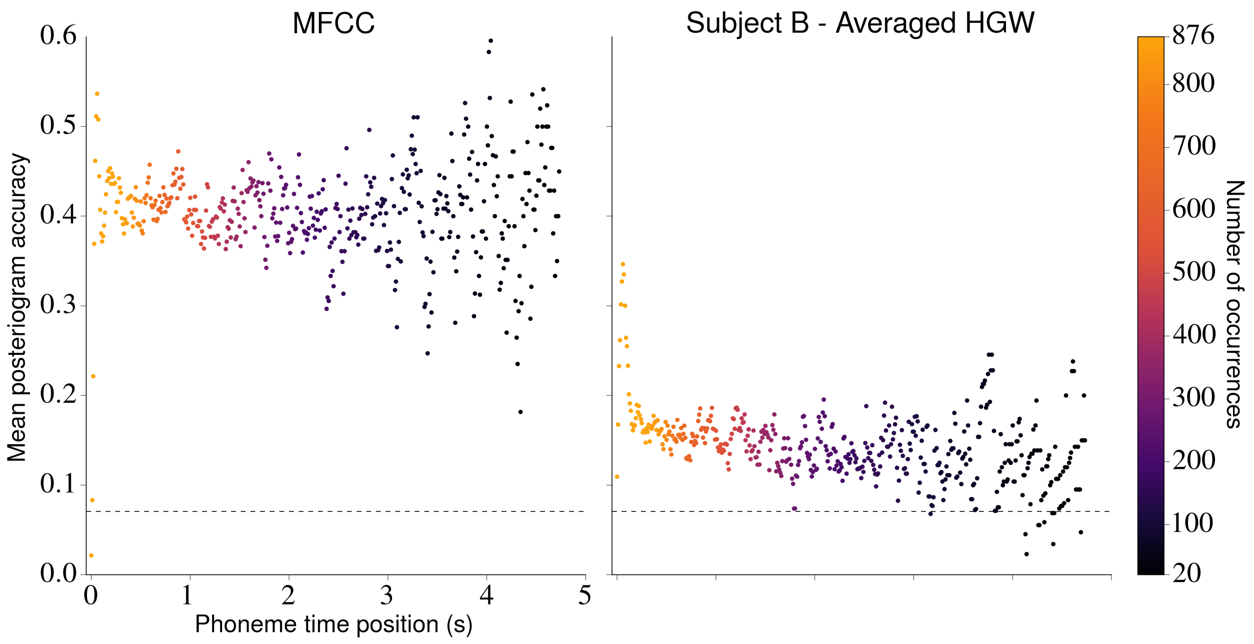


Figure 2.9: The effect of phoneme time position on posteriogram accuracy. Utterance onsets occur when a non-silence phoneme occurs after a silence duration lasting 500 ms or longer. The mean posteriogram accuracies were computed using estimated posteriograms associated with each stimulus in the Gump set generated with MFCCs (left) and averaged HGWs from subject B (right). Each dot represents the mean posteriogram accuracy associated with a phoneme time position, and the color of the dot indicates how many utterances contained a non-silence phoneme at that position. Phoneme time positions that were present in fewer than 20 of the stimuli and all silence phonemes were excluded from the figure. The horizontal dashed line in each plot depicts chance posteriogram accuracy. The apparent heteroscedasticity in each plot is most likely caused by the decreased number of occurrences of non-silence phonemes in the latter part of the utterances (because the utterances differ in duration), which led to less confident predictions of the mean accuracy at those time points. Testing with mixed effects logistic regression models revealed statistically significant negative relationships between phoneme time position and classification accuracy for neural features but not for MFCC features.

the associated data set to assess the effect that phoneme time position had on classification accuracy (Baayen et al., 2008; Jaeger, 2008). In addition to the fixed effect of phoneme time position, random intercepts and slopes were utilized for each stimulus (Barr et al., 2013), which allowed the fits for each stimulus to vary in terms of overall classification accuracy and extent to which accuracy changes as a function of phoneme time position.

For MFCC features, we did not find evidence that phoneme time position influenced classification accuracy ($\beta = 0.051$, $p = 0.225$). Here, β is the regression coefficient for the phoneme time position variable. However, for averaged HGWs from subject B, the analysis revealed a significant effect in which accuracy diminished as a function of phoneme time position ($\beta = -0.792$, $p < 10^{-10}$). After performing this analysis using the remaining feature types described in Section 2.4.2, we observed a similar negative effect for every evaluation involving neural features (each $\beta < -0.4$, $p < 10^{-5}$) and no statistically significant effect for the chance evaluation ($\beta = -0.589, p = 0.085$).

2.4.4 Speaker gender effects

In ASR research, it has been shown that the gender of a speaker affects the features used to train a speech recognition system, which can ultimately affect the system's ability to decode speech from speakers of the opposite gender (Abdulla and Kasabov, 2001). Because of this, we assessed the effect that speaker gender had on the performance of our NSR system. We performed separate evaluations on the 191 Gump stimuli produced by the male speaker and the 191 produced by the female speaker. We also performed an evaluation using 191 Gump stimuli chosen from both genders. These three evaluations used 10-fold cross-validation schemes with attempts to maintain homogeneity between folds (as described in Section 2.3.1). We performed two additional evaluations by training on 90% of the stimuli from the male speaker and testing on all of the stimuli from the female speaker and then repeating this evaluation with the gender roles (training versus testing) swapped. All of these evaluations were performed using single-trial and averaged HGWs for each subject, concatenated HGWs, and MFCCs.

For MFCC features, this analysis revealed significant differences between evaluations for each metric (Welch's ANOVA, $p < 0.01$). For each metric other than the estimation PER metric, post-hoc analyses revealed that the two evaluations involving training on the stimuli from one speaker and testing on the stimuli from the other speaker performed significantly worse than the other three evaluations (Welch's t-test with Bonferonni correction, $p < 0.001$).

For the neural features, most analyses revealed no significant differences between evaluations for each metric (Welch's ANOVA, $p > 0.01$). There were three exceptions: (1) the estimation PER for single-trial HGWs from subject A (Welch's ANOVA, $p = 8.28 \times 10^{-3}$), (2) the estimation PER for averaged HGWs from subject A (Welch's ANOVA, $p = 1.30 \times 10^{-3}$), and (3) the estimation posteriogram accuracy for single-trial HGWs from subject B (Welch's ANOVA, $p = 6.57 \times 10^{-3}$). Overall, the differences between evaluations for each neural feature were negligible compared to the differences observed for acoustic features.

2.5 Discussion

Using relatively simple feature extraction techniques and model components, our NSR system was able to perform, to a limited extent, continuous speech decoding using neural signals. The novel results presented in this work quantitatively indicate that spatiotemporal modeling and ASR techniques, specifically language modeling and Viterbi decoding, can be used to improve phoneme recognition when using neural response features and continuous speech stimuli.

Feature selection had a significant impact on the performance of our NSR system. Unlike ASR, which contains well-established representations of audio waveforms such as MFCC vectors, the ideal representations of cortical surface recordings for the purpose of decoding speech remain unknown. We used HGWs as a relatively simple way to explore this realm of potential representations, guided by our hypothesis that including temporal information in the feature vectors would improve decoding in our system. The results of the feature window grid searches suggested using information contained in the neural responses occurring between 0–250 ms after an acoustic stimulus (within a continuous context) for maximum discriminative ability. However, the certainty of this conclusion is limited by the HGW parameterization constraints, the linearity assumption implicit in the LDA model used to evaluate the HGWs, and the relatively small amount of data used during the grid search. One reason why the HGW parameterization constraints are particularly troublesome arises from the fact that previous research has shown that different sub-populations of cortical neurons in the STG have different response properties (Steinschneider et al., 2011), which suggests that forcing the same HGW parameters to be used for each electrode restricts the capability of HGWs to accurately represent the neural activity. It is also possible that the temporal dynamics are better represented implicitly within the models, through techniques such as sub-phone modeling (Jurafsky and Martin, 2009; Gold et al., 2011) or recurrent neural network (RNN) modeling (Elman, 1990), than explicitly in feature vectors. Additionally, despite the fact that power in the high gamma band has been used effectively in related research, the results of other research efforts indicate that it might be beneficial to evaluate the efficacy of using measures of the raw ECoG signal, power in other frequency bands, and phase information in feature vectors (Luo and Poeppel, 2007). Furthermore, previous research suggests that speech sequence statistics are encoded in the human temporal cortex (Leonard et al., 2015), suggesting that modeling the phonotactic information encoded directly in the neural signals can potentially be incorporated into an NSR system to improve performance. Although representations that are more powerful than these simple HGWs could be uncovered in future research, our results emphasize the importance of modeling spatiotemporal dynamics of neural activity when attempting to discriminate between responses evoked by varying continuous stimuli (at least within the context of speech perception analysis).

As described in Section 2.4.2, the use of HGWs over HGSs and the use of language modeling and decoding tended to improve performance. HGWs consistently provided improvements when compared to HGSs, but the use of a phonemic LM and Viterbi decoding typically provided improvements only for the PER and confusion accuracy metrics and not for the posteriogram accuracy metric. The similarity in the posteriogram accuracy values for estimation and decoding results suggest that the basic phoneme priors used in the estimation results (as described in Section 2.3.2) were as effective at frame-by-frame classification as the phonemic LM used in the decoding results. Altogether, these results indicate that temporal smoothing of the phoneme likelihoods is the primary benefit of incorporating a phoneme-level LM and performing Viterbi decoding. This claim is also supported by the sensitivity of the PER metric to temporal jitter in the predicted phoneme sequence (and the fact that decoding PERs were more favorable than estimation PERs), the apparent smoothing in many of the decoded posteriograms (such as the one depicted in Fig. 2.6), and the similarity between the estimation and decoding confusion matrices (such as the ones depicted in Fig. 2.7). From comparisons between the estimated and decoding confusion matrices in Fig. 2.7, the decoding techniques also seem to reduce the confusability of non-silence phonemes with /sp/. This is most likely a result of the impact that the high occurrence frequency of /sp/ had on the priors described in Section 2.3.2 used when computing the phoneme posteriors. We anticipate that the use of a word-level LM would have a much more significant impact on the differences between estimation and decoding results because predicted phoneme sequences would be restricted to those that comprise word sequences. Additionally, future research could assess the effect that stimulus length has on decoding performance; because the Viterbi parameters affect decoded sequence length, decoding performance could be improved if stimuli of similar lengths were used throughout the development of an NSR system.

Averaging across stimulus presentations typically lessened the negative impact that large trial-by-trial variabilities in the neural responses had on our LDA model. Also, performance was improved using combined features across multiple subjects, which implies that the system could be limited by the spatial resolution of the ECoG arrays, the cortical response properties of individual subjects, or the inherent noise present in recorded ECoG signals. The results using concatenated feature vectors illustrate the upper limits on system performance and the amount of information available in recorded neural signals given the current physical and methodological limitations of our system. However, averaging and combining features across multiple subjects are relatively infeasible approaches for a speech prosthetic application. Future research efforts could explore alternative modeling and preprocessing techniques to obtain more accurate and less variable results using single-trial data from a single subject.

As expected, the MFCC features proved more effective than any of the neural features. However, we were able to observe similarities between the confusion matrices generated using neural and acoustic data (as shown in Fig. 2.7). In both cases, confusions typically occurred amongst stops, affricates, and fricatives or amongst the vowels, although for neural data the vowels were more confusable with the nasals and approximants than for acoustic data. These confusion matrices also suggest that prediction accuracy for stops was similar for neural and acoustic features. Additionally, for both feature types, our system was extremely effective at predicting silence, as made evident by the large phoneme confusion values for /sp/ in these confusion matrices.

We found a negative correlation between the time position of a phoneme in an utterance and our system's ability to correctly predict that phoneme when neural (and not acoustic) features are used (as discussed in Section 2.4.3). One factor that could be contributing to this observation is the existence of transient neural responses to acoustic onsets that might encode phonetic information in fundamentally different ways than sustained responses (Tiitinen et al., 2012; Okamoto and Kakigi, 2014). This correlation could also indicate that response patterns evoked by a phoneme, which can last hundreds of milliseconds, are overlapping with response patterns of subsequent phonemes, resulting in observed responses that grow increasingly complex as an utterance progresses. Another possibility is that the cortical responses used in our analyses also contain representations of higher-order information related to the perceived speech, such as word identity (Cibelli et al., 2015) or phonotactic information (Leonard et al., 2015), which could affect the accuracy of the phoneme likelihood model. Because the observed degradation of prediction quality over time is particularly problematic for continuous speech decoding approaches, attempts to directly model these effects could lead to performance improvements in future iterations of our NSR system.

As described in Section 2.4.4, we showed that the gender of the speakers that generated the stimuli typically had no effect on the performance of our system when using neural features. As expected, speaker gender did have a significant effect on the system's performance when using MFCCs. Because gender is one of the most influential sources of speaker-attributed acoustic variability during speech production (Abdulla and Kasabov, 2001), we conclude that speaker identity does not significantly affect our system when using neural features. This conclusion is consistent with the theory that phonetic information is encoded more strongly in the STG than other information that is more variable between genders (such as fundamental frequency) (Mesgarani et al., 2014) and suggests that data from multiple speakers can be used to effectively train an NSR system.

To further assess the potential of using our NSR system in a speech prosthetic application, we can repeat our analyses using neural signals recorded during covert speech. Research groups have shown that covert and overt speech share partially overlapping neural representations in the auditory cortex and that it is feasible to reconstruct continuous auditory speech features from ECoG data recorded during a covert speech task (Tian and Poeppel, 2010; Martin et al., 2014). A future NSR system capable of intelligibly decoding covert speech could lead to the development of a speech prosthetic that allows impaired individuals to communicate more naturally with others. In addition, it could be beneficial to repeat our analyses using speech production tasks and neural activity from motor areas. Two research groups have reported favorable results by decoding produced speech using ECoG signals recorded in the motor cortex, although these groups did not perform continuous speech decoding (Kellis et al., 2010; Mugler et al., 2014). Also, by expanding on the approaches described in this work, stimuli containing speech from multiple speakers simultaneously could be used to add to the current knowledge of how the brain handles encoding of speech information in a multi-speaker setting (Mesgarani and Chang, 2012) and to further ongoing research efforts aimed at gaining insights applicable to ASR systems that operate in multi-speaker environments (Chang et al., 2015). Future NSR research should also include comparisons that address whether or not a discrete-state decoder (such as our system) that predicts sequences of speech tokens can effectively leverage language modeling and probabilistic decoding to increase performance over continuous-valued reconstruction methods that predict acoustics (such as spectrograms) (Pasley et al., 2012).

The progress described in this work is primarily a proof-of-concept and should provide useful insights for future research in the field of NSR. The relatively simple model components and feature representations used in our system leave much room for improvement. For example, one of many recent advances in the ASR field has shown that modeling the spatiotemporal dynamics of the feature space non-linearly using deep recurrent neural network models can significantly improve decoding performance over more traditional methods (Graves et al., 2013). The PERs reported in these studies are much lower than what we achieved with our system when using acoustic features, which implies that the incorporation of more sophisticated models from modern ASR systems could improve the performance of our NSR system. In addition, we intend to use word-level language modeling and decoding in future iterations of our system to make it more suitable for speech prosthetic applications.

Chapter 3

Real-time classification of auditory sentences using evoked cortical activity in humans

Disclaimer: This chapter is a direct adaptation of the following article:

D. A. Moses, M. K. Leonard, and E. F. Chang, 2018. Real-time classification of auditory sentences using evoked cortical activity in humans. *Journal of Neural Engineering*, 15(3). doi: 10.1088/1741-2552/aaab6f.

Personal contributions: I developed and tested the real-time system, performed all data collection and analyses, and wrote the original draft of the manuscript.

Note: A supplementary video from this article was not included here.

3.1 Abstract

Recent research has characterized the anatomical and functional basis of speech perception in the human auditory cortex. These advances have made it possible to decode speech information from activity in brain regions like the superior temporal gyrus, but no published work has demonstrated this ability in real-time, which is necessary for neuroprosthetic brain-computer interfaces. Here, we introduce a real-time Neural Speech Recognition (*rtNSR*) software package, which was used to classify spoken input from high-resolution electrocorticography signals in real-time. We tested the system with two human subjects implanted with electrode arrays over the lateral brain surface. Subjects listened to multiple repetitions of ten sentences, and *rtNSR* classified what was heard in real-time from neural activity patterns using direct sentence-level and HMM-based phoneme-level classification schemes. We observed single-trial sentence classification accuracies of 90% or higher for each subject with less than 7 minutes of training data, demonstrating the ability of *rtNSR* to use cortical recordings to perform accurate real-time speech decoding in a limited vocabulary setting. Further development and testing of the package with different speech paradigms could influence the design of future speech neuroprosthetic applications.

3.2 Introduction

Recent work has characterized the specific functional roles of the human superior temporal gyrus (STG) and neighboring brain areas in speech perception and language understanding (Boatman et al., 1997; Binder et al., 2000; Canolty et al., 2007; Hickok and Poeppel, 2007; Rauschecker and Scott, 2009; Mesgarani et al., 2014). While subjects are listening to spoken speech, neural activity in this region can be used to decode and reconstruct speech information, including spectrotemporal acoustic properties (Pasley et al., 2012; Yang et al., 2015; Leonard et al., 2016) and phoneme sequences (Moses et al., 2016). Previous work has implemented real-time systems capable of mapping sensorimotor activations using spectral decomposition of neural signals (Cheung and Chang, 2012), using transcribed stimuli to generate neural encoding models (as opposed to decoding models) of segmental speech (e.g. phonemes) (Khalighinejad et al., 2017), decoding isolated phonemes from brain activity (Leuthardt et al., 2011), and detecting speech production onsets and offsets from cortical responses (Kanas et al., 2014). However, to the best of our knowledge no published work has demonstrated real-time classification of phoneme sequences or entire sentences from neural signals, which would have practical applications in speech neuroprostheses.

In this work, we developed and tested a real-time Neural Speech Recognition (*rtNSR*) software package. As defined in our previous work, we use the term neural speech recognition to refer to performing speech decoding using neural responses as features (Moses et al., 2016). The *rtNSR* package contains real-time code capable of presenting visual and acoustic stimuli, processing acquired neural signals, training probabilistic models, performing classification and decoding, and storing data and metadata. Our primary goal in this work was to perform an initial assessment of the capabilities of *rtNSR* using a relatively simple sentence prediction task. In this task, subjects listened to multiple presentations of ten pre-recorded spoken sentences. During these stimulus presentations, cortical activity is obtained in real-time via electrocorticography (ECoG) arrays and used in one of two classification schemes to predict the identity of the stimulus that the subject just heard. The results of this study indicate that *rtNSR* is capable of accurately decoding single-trial speech events in real-time, demonstrating its viability as a platform for an assistive speech application.

3.3 Methods

3.3.1 Subjects

The two subjects (A and B) who participated in this study were human epilepsy patients undergoing treatment at the UCSF Medical Center. To aid clinicians in localizing seizure foci, two 128-channel ECoG arrays with 4 mm center-to-center electrode spacing (PMT corp.) were surgically implanted on the cortical surface of each subject. Both subjects had unilateral coverage over the right hemisphere that included the STG. MRI brain reconstructions with electrode locations were generated for each subject using the open source *img_pipe* package (see Fig. 3.5) (Hamilton et al., 2017).

Both patients gave their informed consent to be a subject for this research prior to surgery. The research protocol was approved by the UCSF Committee on Human Research.

3.3.2 Speech stimuli

In each experimental task, the subject listened to multiple repetitions of ten phonetically transcribed speech stimuli from the Texas Instruments/Massachusetts Institute of Technology (TIMIT) dataset (Garofolo et al., 1993). In each stimulus, a single speaker produces a single sentence. We trimmed silence from each end of each stimulus sound file prior to running the tasks. The TIMIT label, sentence transcription, and duration of each stimulus are provided in Table 3.1.

We converted each speech sound label specified in the phonetic transcriptions to one of the 37 phonemic labels used in this work. This set of phonemic labels, which is provided in Table 3.2, is comprised of 36 phonemes from the ARPABET and /sp/, a silence phoneme used to label non-speech data points.

Table 3.1: Information about each stimulus.

TIMIT label	Sentence transcription	Duration (s)
fcaj0_si1479	Have you got enough blankets?	1.108
fcaj0_si1804	It had gone like clockwork.	1.540
fdfb0_si1948	He moistened his lips uneasily.	1.527
fdxw0_si2141	It was nobody's fault.	1.161
fisb0_si2209	"A bullet," she answered.	1.508
mbbr0_si2315	Junior, what on earth's the matter with you?	1.679
mdlc2_si2244	Nobody likes snakes.	1.301
mdls0_si998	Yet they thrived on it.	1.000
mjdh0_si1984	And what eyes they were.	1.048
mjmm0_si625	A tiny handful never did make the concert.	2.106

Table 3.2: The phonemic labels used in this work and their respective categorizations.

Category	Phoneme
Silence	sp
Stop	b d g p t k
Affricate	jh
Fricative	f v s z sh th dh hh
Nasal	m n ng
Approximant	w y l r
Monophthong	iy aa ae eh ah uw ao ih uh er
Diphthong	ey ay ow oy

3.3.3 Real-time processing setup

An overview of the real-time stimulus prediction system is depicted in Fig. 3.1. The capital letter labels in this figure correspond to the data flow steps during each stimulus presentation (each trial) in each task block. At the start of each trial, a Linux workstation (64-bit Ubuntu 14.04, Intel Core i7-4790K processor, 32 GB of RAM) implementing *rtNSR* plays one of the stimuli to the subject (A-B). Simultaneously, the implanted ECoG arrays record cortical local field potentials at 256 cortical sites, which are processed in the data acquisition (DAQ) rig (C). Within the DAQ rig, the ECoG signals are amplified and quantized at 3051.76 Hz using a pre-amplifier (PZ2, Tucker-Davis Technologies) and preprocessed using a digital signal processor (RZ2, Tucker-Davis Technologies). Before the ECoG signals are preprocessed, they are stored on the rig along with the time-aligned audio waveform. During preprocessing, the signals are notch filtered at 60, 120, and 180 Hz to reduce line noise. Next, each channel is band-passed at 70–150 Hz, squared, and smoothed using a low-pass filter at 10 Hz to extract power in the high gamma band. High gamma power was used because previous

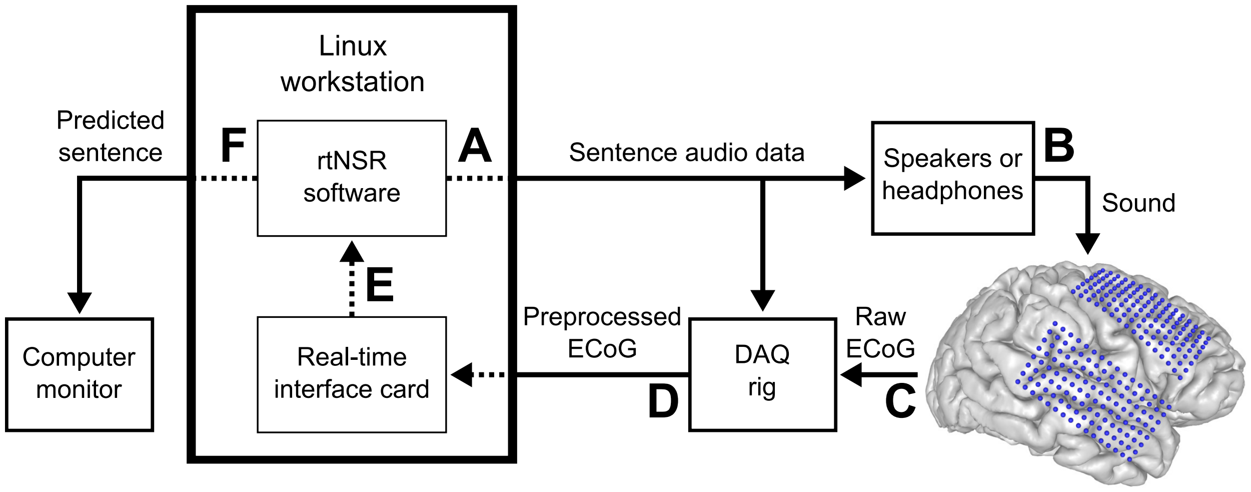


Figure 3.1: A schematic depiction of the real-time stimulus prediction system with the letters A-F denoting the flow of information through the system. A Linux workstation implementing *rtNSR* plays the stimuli to the subject during ECoG data collection (A-B). The raw ECoG signals are amplified, preprocessed, and synchronized with the audio data in the data acquisition (DAQ) rig (C). The preprocessed ECoG signals are streamed to the workstation through a real-time interface card (D). The *rtNSR* software acquires the signals from the card, processes them, and uses them to perform sentence classification (E). Sentence predictions are displayed on a computer monitor (F). The MRI brain reconstruction for subject A is shown here with electrode locations depicted as blue dots. Electrode coverage was similar for subject B (see Fig. 3.5).

research has shown that activity in this band strongly correlates with multi-unit activity (Crone et al., 1998) and is associated with important speech features (Pasley et al., 2012; Mesgarani et al., 2014; Moses et al., 2016). These high gamma signals are then decimated to 98.44 Hz and streamed to the Linux workstation using a real-time interface card (PO8e, Tucker-Davis Technologies) where they are processed in *rtNSR* and saved to disk for offline analyses (D-E). Further discussion of preprocessing considerations and feature extraction are available in Section 3.5.

Within *rtNSR*, the signals acquired from the real-time card are normalized by z-scoring the data for each channel using a 30-second sliding window. These z-score values are clipped to lie within the range of $[-2, 2]$ to mitigate signal artifacts caused by epileptic activity, channel noise, or other factors. If a trained model is available, then, immediately after the stimulus presentation, signals from relevant channels are used as features in this model to predict which stimulus was just presented to the subject (detailed descriptions of the channel selection and modeling procedures are given in Section 3.3.6). The stimulus prediction and updated running classification accuracy measures are displayed on a monitor (F).

3.3.4 *rtNSR* design

Our *rtNSR* system is implemented in the Python programming language (Python Software Foundation, 2010) and is designed for flexible and efficient real-time neural signal modeling and speech decoding. Based on the software pipelining implementation technique (Lam, 1988), *rtNSR* uses multiple data processing elements that run in parallel as individual processes. Typically, each of these processes obtains inputs from one or more separate processes (via software pipes or shared memory buffers), performs a specific task with or manipulation on the inputs, and sends outputs to one or more other processes. Each process is defined as a sub-class of a parent real-time process class implementing general methods for real-time processing (including data sharing and process setup methods). *rtNSR* contains many of these single-purpose process classes, such as a process that reads streaming data from the real-time interface card and a process that performs sliding window normalization. This highly modularized software architecture allows for individual steps in the real-time processing workflow to be interchanged and rearranged with relative ease while leveraging the computational efficiency associated with pipelining and parallelization. For example, during real-time simulations performed offline for debugging and system evaluation, we simply replaced the real-time card reader process in the data processing workflow with a process that loads and streams out pre-recorded neural data. A block diagram depicting the *rtNSR* components and data flow used during the real-time experiments is provided in Fig. 3.2.

3.3.5 Experimental task blocks

For subject A, we collected a total of 300 stimulus presentations (30 for each stimulus) across a total of 4 task blocks. For subject B, we collected a total of 250 stimulus presentations (25 for each stimulus) across a total of 3 task blocks. At the start of each block, a 1-second beep is played to signal the start of the task. This sound triggers an audio onset detector in the preprocessor to inject a start token into an arbitrarily chosen recording channel. The sentences are then presented with a constant onset-to-onset time interval. As a result, *rtNSR* can easily keep track of which neural data points are associated with each stimulus presentation (see Section 3.5 for further discussion on stimulus timing). Within each block, we randomized the stimulus presentations while ensuring that each stimulus was presented an equal number of times.

In each task block, the onset-to-onset interval was approximately 2.57 seconds, the stimuli were presented aurally via loudspeakers, and the subject was not able to see the real-time stimulus classifications. However, in the final block for subject A, the onset-to-onset interval was approximately 5.14 seconds, the stimuli were presented using headphones, and the subject was instructed to view the real-time sentence classifications and respond with either a "thumbs up" or a "thumbs down" to indicate if the prediction matched what was heard through the headphones. The extra time in the onset-to-onset interval for this block was not used during modeling and was only included to allow the subject to respond before the onset of a new sentence.

3.3.6 Stimulus classification schemes

Stimulus classification models were trained for each subject using data collected during experimentation. Each time a model was trained, the collected data were first analyzed to identify which channels should be considered relevant to speech perception processing

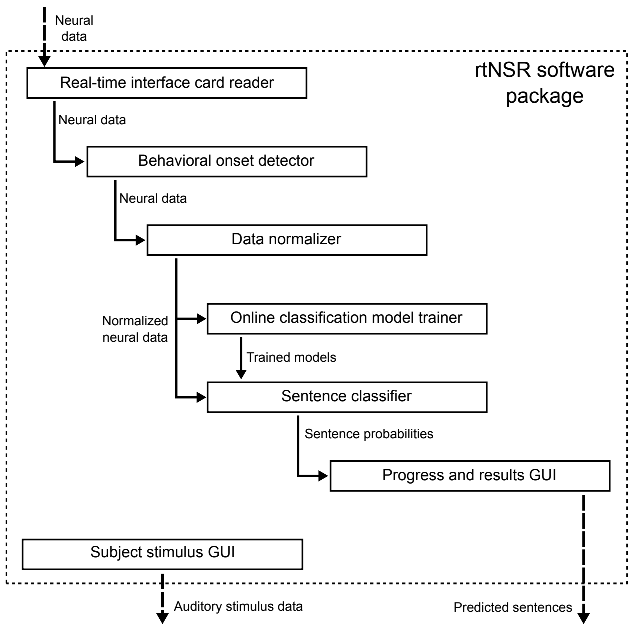


Figure 3.2: A schematic depiction of the *rtNSR* implementation used during experimentation. The solid rectangles represent real-time process classes and the arrows represent data that is passed between processes. The *Real-time interface card reader* process reads neural data streamed from the real-time interface card. These data points are passed to the *Behavioral onset detector* process, which detects a one-time injected onset token that signifies the start of the task (see Section 3.3.5). The neural data are then passed to the *Data normalizer* process, which performs sliding window normalization and magnitude clipping. The normalized neural data are passed to the *Sentence classifier* process, where the data are used to perform sentence classification. This process outputs sentence probabilities to the *Progress and results GUI* process, which extracts the most likely sentence from each of these sentence probability vectors and displays each predicted sentence on a monitor. When using the direct classification scheme, the *Online classification model trainer* process also obtains the normalized neural data, performs model training and relevant electrode selection in real-time, and passes trained models (with relevant electrode numbers) to the sentence classification process (see Section 3.3.6). Throughout the task, the *Subject stimulus GUI* process controls auditory presentation of the sentence stimuli to the subject.

(Moses et al., 2016). A simple bad channel detector was used to exclude any channels for which 75% or more of the acquired data points had a z-score of 0.25 or less. Afterwards, two data subsets were created: one subset comprised of neural data sampled during sentence perception of each stimulus presentation (30 time points per stimulus presentation) and another similarly constructed subset containing data points sampled during the silence after each sentence. Two-tailed Welch's t-tests were then performed for each channel between the two data subsets. Channels that exhibited a p-value less than 0.001 were considered relevant (significantly modulated by the presence of auditory speech stimuli) and the remaining channels were excluded during modeling. Applying these procedures to the data acquired before the final testing block resulted in 79 and 122 relevant electrode channels for subjects A and B, respectively (see Fig. 3.5).

We used two types of real-time stimulus classification schemes in our tasks: a “Direct” classification scheme during testing with subject A and an “HMM-based” classification scheme during testing with subject B. As described in Section 3.3.4, we were able to slightly modify the experimentation setup to simulate stored neural data as if it were being obtained in real-time without altering the classification scheme functionality. This enabled us to compute results for each subject using the classification scheme that was not used during real-time testing for that subject. After data collection and these offline simulations, results using both schemes were available for each subject. We used the *scikit-learn* Python package to implement the models employed in both schemes (Pedregosa et al., 2011).

Direct classification scheme

In the direct classification scheme, each stimulus (sentence) was treated as one of ten classes. The feature vectors used during classification were each constructed by concatenating the z-scored high gamma power values for each relevant channel at each time point during a stimulus presentation. Because the stimuli varied in duration, some of the neural data obtained during the silence periods after a stimulus presentation were included in the feature vector associated with that stimulus presentation. The feature vector for each stimulus presentation contained the neural data at each of the $T = 253$ time points associated with that presentation (which spans the 2.57 second time window allotted for each presentation, as described in Section 3.3.5). For example, a stimulus presentation that began at time index t would be associated with a feature vector containing the neural data points for each relevant channel at time indices $\{t, t + 1, \ldots, t + T - 1\}$ (with a length of T times the number of relevant channels) and with a target label equal to the identity of that stimulus.

During model training, we use principal component analysis (PCA) to reduce the dimensionality of the feature vectors to the minimum number of features required to explain at least 99% of the variance. The resulting feature vectors have lengths that are typically around 100 elements (less than 1% of the lengths of the original vectors). These new feature vectors are used to train a linear discriminant analysis (LDA) model implementing the least-squares solution with automatic shrinkage using the Ledoit-Wolf lemma (Ledoit and Wolf, 2004). Once trained, we used these combination PCA-LDA models to classify previously unseen neural responses into one of the ten stimulus labels in real-time. Model training, which typically took 2–5 seconds, was first performed in real-time using all available data for a subject when at least 2 repetitions of each stimulus were presented and subsequently performed prior to starting a new task block and in real-time whenever 10 stimulus presentations had occurred since the most recent training.

HMM-based classification scheme

In the HMM-based classification scheme, each stimulus is represented as a hidden Markov model (HMM), where each hidden state q_t is the phoneme that occurs at time index t for that stimulus and each observed state y_t is the neural feature vector associated with time index t. This classification scheme was inspired by the phoneme decoding results described in (Moses et al., 2016).

For a normal HMM, the joint probability would be

$$p\left(q, y\right) = p\left(q_0\right) \prod_{t=0}^{T-2} p\left(q_{t+1} | q_t\right) \prod_{t=0}^{T-1} p\left(y_t | q_t\right), \tag{3.1}$$

where $q = \{q_0, \ldots, q_{T-1}\}$, $y = \{y_0, \ldots, y_{T-1}\}$, and $T = 253$ (as defined in Section 3.3.6). However, because each presentation of a stimulus uses the exact same audio waveform, the values of q are already known for each stimulus from the phonetic transcriptions of the stimuli. This simplifies the HMM for each stimulus because the values of the hidden states are known. In this scenario, Bayes' theorem can be used to express the conditional probability associated with each simplified HMM as

$$p\left(y|q\right) = \frac{p\left(q, y\right)}{p\left(q\right)} = \prod_{t=0}^{T-1} p\left(y_t|q_t\right). \tag{3.2}$$

For each stimulus presentation, our HMM-based classification scheme uses Eq. 3.2 to estimate $p\left(y|q\right)$ for each of the ten competing simplified HMMs (one per stimulus) and predicts the stimulus that yielded the largest $p\left(y|q\right)$ value. This can be formally expressed as

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} \prod_{t=0}^{T-1} p\left(y_t|q_{t,s}\right) = \underset{s \in S}{\operatorname{argmax}} \sum_{t=0}^{T-1} \log p\left(y_t|q_{t,s}\right), \tag{3.3}$$

where $\hat{s}$ is the predicted stimulus, S is the set of possible stimuli, and $q_{t,s}$ is the phoneme at time index t for stimulus s. We use log probabilities as expressed in the latter part of (Eq. 3.3) for computational efficiency and numerical stability.

Each feature vector y_t contains the z-scored high gamma power values for each relevant channel at the following time indices: $t + \{0, 2, \ldots, 38, 40\}$. This parameterization of the feature vectors resembles the high gamma windows described in previous research (Moses et al., 2016). We used PCA-LDA modeling (as described in Section 3.3.6) to obtain the $p\left(y_t|q_t\right)$ values at each time point. Model training, which typically took 10–20 seconds, was performed using all available data for a subject prior to starting each new task block (classifications were not performed in the first block).

3.3.7 Evaluation methods

We primarily used classification accuracy (the percent of classification attempts that resulted in correct classifications) to evaluate *rtNSR*. We computed classification accuracies for each task block and in a sliding window fashion across the blocks to measure how the accuracy changed over time.

Because duration was highly variable across sentences and could have been used by the classification schemes for improved sentence discrimination, we also assessed how varying T, the number of time points used during modeling of each stimulus presentation (described in Section 3.3.6), affected classification accuracy. We performed offline testing with 21 different values for T that were (roughly) equally-spaced within the range of $[1, 253]$. For both classification schemes, 10-fold stratified cross-validation was used on all the available data for each subject.

To assess the speed of our real-time classification schemes, we measured the amount of time each classification scheme took to perform classifications during offline simulations. For the direct classification scheme, we measured the amount of time required to make each sentence prediction from a concatenated neural feature vector, which was performed every $T = 253$ time points. For the HMM-based classification scheme, we measured the amount of time required to compute the phoneme likelihood values $p\left(y_t|q_t\right)$ at each time point and the amount of time required to perform a sentence classification using the associated phoneme likelihoods every $T = 253$ time points.

3.4 Results

For subject A, we achieved stimulus prediction accuracies of 90% with the direct classification scheme in real-time and 98% with the HMM-based classification scheme during offline simulation after training on 250 stimulus presentations (approximately 11 minutes of training data). For subject B, we achieved accuracies of 90% with the direct classification scheme during offline simulation and 91% with the HMM-based classification scheme in real-time after training on 150 stimulus presentations (approximately 6.5 minutes of training data). Confusion matrices for these results are provided in Fig. 3.6. All observed classification accuracies are depicted in Fig. 3.3.

Fig. 3.4 depicts the effect that varying the number of time points used during classification had on accuracy. When only the first 89 time points (approximately 0.9 seconds) for each trial were used, which is less than the number of time points associated with the shortest sentence, the classification accuracies plateaued at 90% or higher. These results indicate that the classification schemes are relying on more than just sentence length when performing classifications and that highly accurate classification can be performed using neural responses collected during perception of the first $2-3$ words of the sentences.

During offline simulation of the HMM-based classification scheme with subject A, computing the phoneme likelihoods at each time point took on average 2.64 ms ($\sigma = 0.61$ ms, $N = 12650$) and each sentence classification (using the pre-computed phoneme likelihoods) took on average 0.07 ms ($\sigma = 0.01$ ms, $N = 50$). During offline simulation of the direct classification scheme with subject B, each sentence classification took on average 10.23 ms ($\sigma = 3.99$ ms, $N = 100$).

3.5 Discussion

In this work, we have introduced a real-time Neural Speech Recognition (*rtNSR*) software package and demonstrated its ability to perform real-time, single-trial stimulus classification using cortical responses evoked during speech perception. We achieved high classification accuracies after short training intervals using both direct (sentence-level) and HMM-based (phoneme-level) classification schemes. The HMM-based classification scheme exhibited the highest observed accuracy in a single block (98% accuracy with subject A).

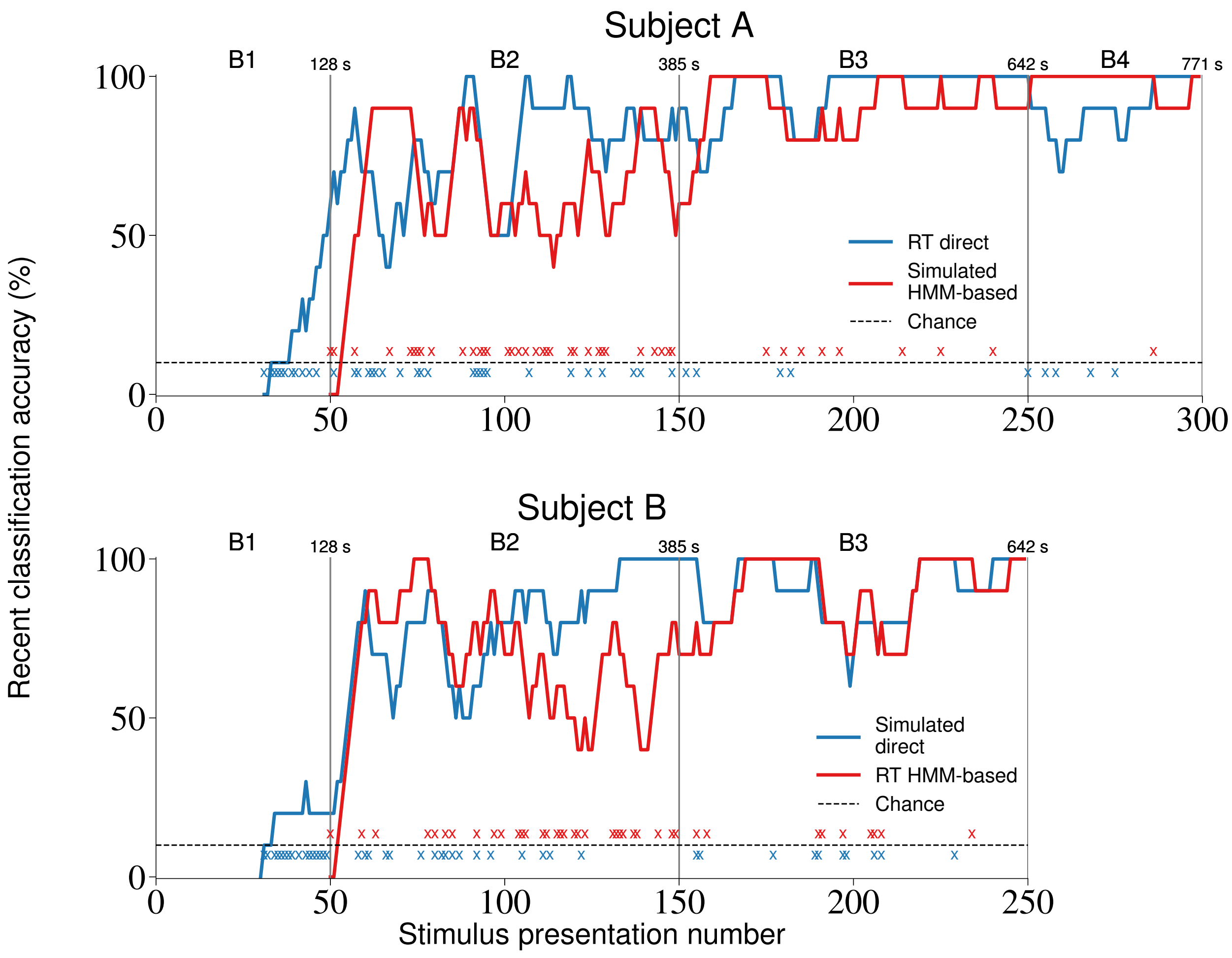


Figure 3.3: Stimulus classification accuracies for each subject, task block, and classification scheme. The colored curves depict, for each stimulus presentation, the percentage of the 10 most recent classification attempts (including the current attempt) that were correct. The blue and red curves represent testing with the direct and HMM-based classification schemes, respectively. Results obtained from real-time testing contain "RT" in the label and those obtained from offline simulations contain "Simulated" in the label. A colored x marker indicates a trial that was incorrectly classified with the associated classification scheme. The task blocks are labeled (with "B" followed by the block number) and separated by vertical lines. The total duration of recorded data at the end of each block is given above these vertical lines (rounded to the nearest second). Chance accuracy (10%) is depicted as a horizontal dashed line. These plots exhibit that *rtNSR* is able to achieve high real-time classification accuracies after short training intervals.

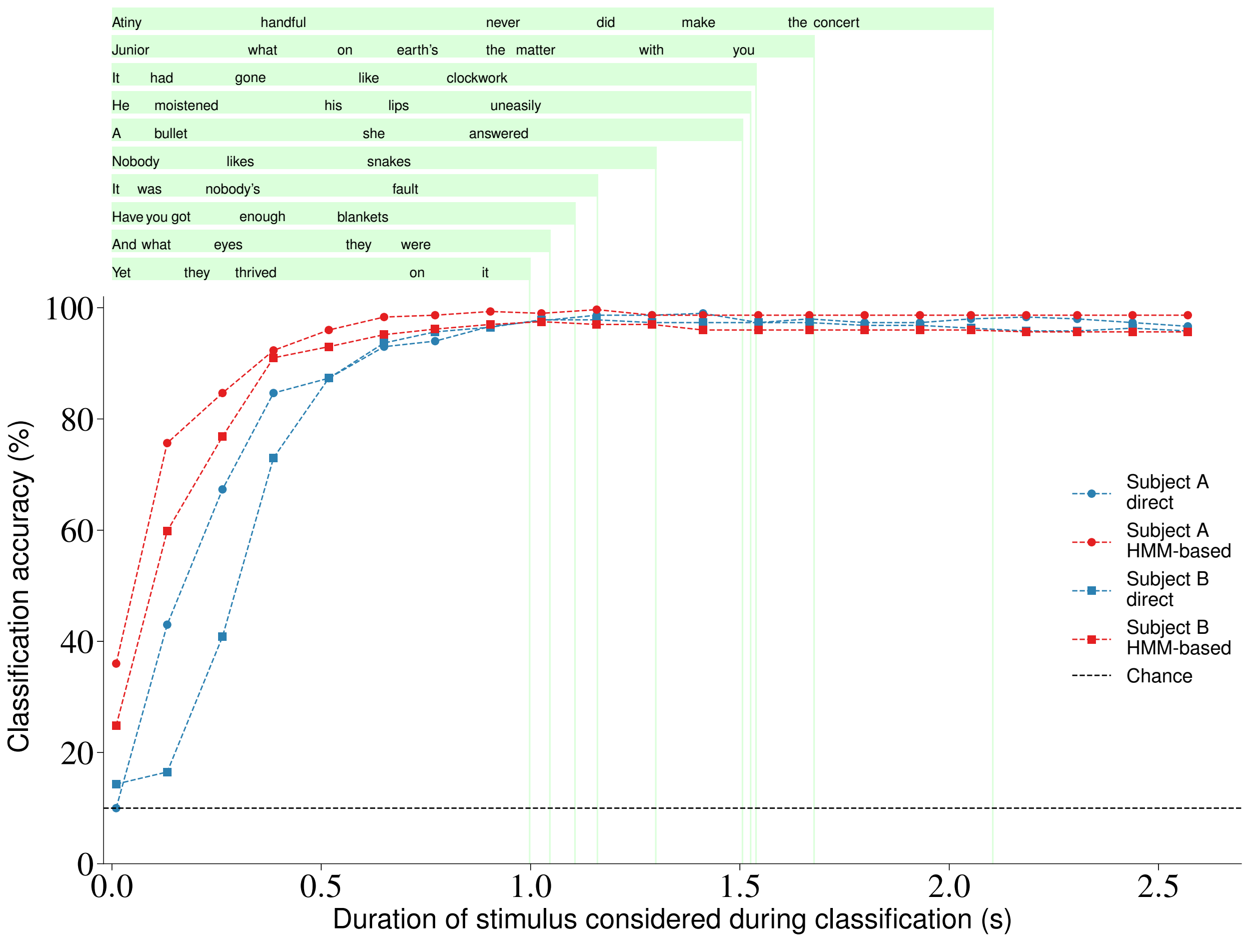


Figure 3.4: The effects of varying the duration of each stimulus presentation used during classification. For each subject and classification scheme, the corresponding colored dots or squares and dashed line depict the classification accuracies associated with the considered stimulus durations. The transcriptions for each sentence are shown above the plot. The left boundary of each word is aligned to the time at which that word begins within the sentence audio files. The light green rectangles and vertical lines indicate the full time span and offset time, respectively, for each sentence. Chance accuracy (10%) is depicted as a horizontal dashed line. These accuracy curves indicate that both classification schemes are able to leverage information in the neural signals during perception of the initial 0.75 seconds of each sentence to accurately discriminate between sentences.

We showed that neural activity collected during perception of naturally spoken sentences could be used directly for classification without including acoustic, phonetic, or any other stimulus information (other than sentence identity) during modeling (with the direct classification scheme). We also showed that similar performance could be achieved with a more sophisticated classification approach that involved modeling the neural representations of phonemes (with the HMM-based classification scheme). Additionally, we demonstrated that the performance of our system did not rely on sentence length, a trivial stimulus feature, since peak classification accuracies were obtained using only a subset of time points associated with each trial that was smaller than the duration of the shortest sentence in the task. Finally, we showed that *rtNSR* was able to perform real-time classifications quickly; on average, the direct classification scheme only required 10 ms every 2.57 seconds (the stimulus time window duration) to perform a classification and the HMM-based classification scheme only required less than 3 ms every 10.16 ms (the sampling interval) to compute phoneme likelihoods at each time point and a negligible amount of time to make a sentence prediction from the phoneme likelihoods.

Our results serve as a proof-of-concept that *rtNSR* is capable of performing speech classification from neural signals in real-time. We built the *rtNSR* system to have a modular architecture in which individual components can be improved or replaced with task-specific and optimized implementations for future applications. For example, the high gamma power estimation algorithm implemented on the DAQ rig can be replaced with digital filters in *rtNSR* that directly approximate the high gamma analytic amplitude, a representation of high gamma activity that has been used in previous speech-related research (Canolty et al., 2007; Pasley et al., 2012; Moses et al., 2016). Also, the sentence classification process can be replaced by a process implementing a more sophisticated classification model, such as a recurrent neural network classifier. In addition, the software's robust task design and execution capabilities make it amenable to a variety of task paradigms, including isolated word or continuous speech production or perception tasks, visual stimulus presentation tasks, and covert speech tasks. Through augmentation of the system's data acquisition and feature extraction functionality, it can also be deployed in applications involving alternative types of neural signal acquisition, such as via electroencephalography or microelectrode arrays.

For an initial evaluation of our system, we used a relatively simple sentence classification task with only 10 unique stimuli. Although the observed classification accuracies were very high in this example task, demonstrating our ability to learn the relationship between auditory speech stimulus features and neural activity recorded with ECoG in real-time, further testing is needed to determine how well the classification schemes scale as the number of stimuli increases. We expect the HMM-based classification scheme to scale more favorably than the direct classification scheme because it can take advantage of shared phonemic content across the stimuli and can predict stimuli that were not presented during training. However, it is also possible that an increase in the variety of coarticulation contexts and other sources of variability in the stimuli will negatively affect accuracy if they are not explicitly considered during modeling.

We established that one trivial stimulus feature (duration) did not drive classification performance, but there are other potential features that may have impacted accuracy. In this task, nine speakers produced the ten stimuli, resulting in a large degree of variability in the speaker-dependent acoustic properties of the stimuli that may have been leveraged by the classification schemes. When analyzing the sentence confusions observed during classification (see Fig. 3.6), we did not find evidence that speaker identity was driving our classifiers in this task. However, it is possible that in experiments involving a larger set of sentences from relatively few speakers the direct classification scheme would be more susceptible to relying on speaker identity than the HMM-based classification scheme, since the latter uses phoneme models that do not incorporate stimulus identity information while being trained to discriminate between phonemes. Future work using a wider variety of stimuli with multiple speech samples produced by each speaker could address the effects of this type of information on classification performance.

In future work, we plan to expand the HMM-based classification scheme into a real-time continuous speech decoder that uses language modeling and Viterbi decoding (similar to a real-time version of the system described in (Moses et al., 2016)). The performance achieved in this work using phoneme modeling with naturally spoken sentences (as opposed to isolated words or syllables) is a promising proof-of-concept for potential continuous decoding applications. Unlike our task, a real-time continuous decoding application should not rely on explicit stimulus timing, although precise transcriptions of the stimuli would still be required for model training. The methods described in this work could also be applied to real-time experimental paradigms in overt and covert speech production tasks guided by existing offline speech decoding research efforts (Kellis et al., 2010; Pei et al., 2011; Denby et al., 2010; Martin et al., 2014; Mugler et al., 2014; Herff et al., 2015; Martin et al., 2016).

After further development of *rtNSR*, our goal is to deploy the system as part of a speech prosthesis that restores communicative capabilities to individuals diagnosed with locked-in syndrome or other impairments. Locked-in patients typically have little to no voluntary muscle control but retain cognition and awareness (American Congress of Rehabilitation Medicine, 1995; Laureys et al., 2005; Bruno et al., 2011; Rousseau et al., 2015; Vansteensel et al., 2016). Although methods exist to provide basic communicative capabilities to locked-in patients (Spüler et al., 2012; Sellers et al., 2014; Mainsah et al., 2015; Vansteensel et al., 2016) and are associated with increases in patient-reported quality of life (Bruno et al., 2011; Rousseau et al., 2015), these approaches often involve tedious and difficult to learn procedures such as selecting characters one at a time at rates less than 10 characters per minute (typing rates are typically more than 175 characters per minute in healthy individuals). Development of a device capable of directly interpreting intended speech from neural activity could result in significant improvements to the speed and naturalness of assistive speech technology and, as a result, the quality of life for impaired patients. Existing brain-computer interface (BCI) research has shown that ECoG signals can be successfully used in real-time motor control applications (Leuthardt et al., 2006; Hotson et al., 2016), and the classification accuracies observed in this task using ECoG are similar to or higher than those exhibited in these approaches (although direct performance comparisons may not be possible due to fundamental differences in task designs and constraints). Our system's modular real-time framework allows for incorporation of feedback and subject adaptation, important components in closed-loop BCIs that will most likely be beneficial in future speech prostheses. Given the performance exhibited by *rtNSR* in this work and its capacity for expansion, we are confident in its ability to serve as a platform for the design and implementation of the proposed speech prosthetic device.

3.6 Supplementary data

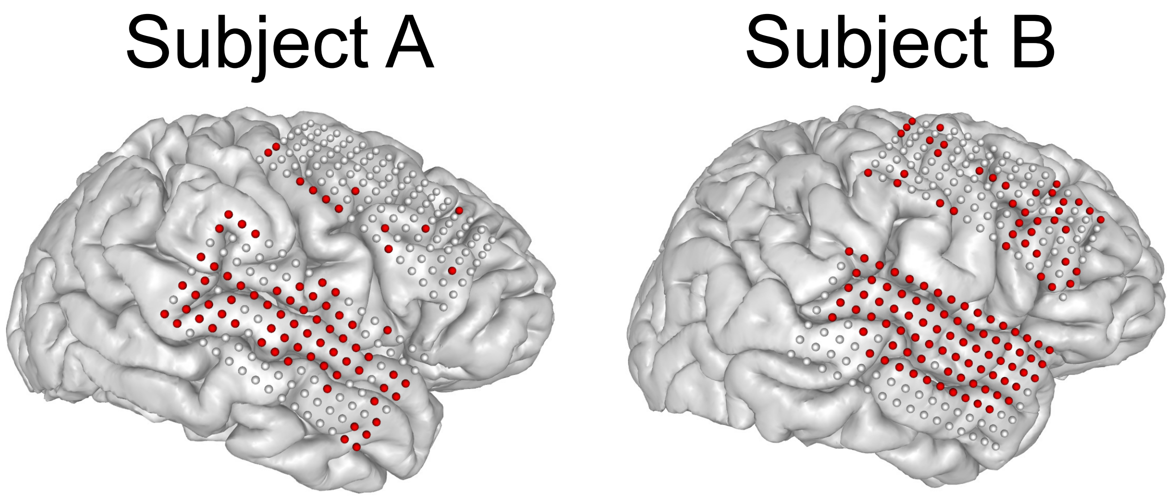


Figure 3.5: 3-D MRI brain reconstruction, electrode coverage (white and red dots), and relevant electrodes (red dots) for each subject. The depicted relevant electrodes were determined using the data acquired prior to the final testing block for each subject.

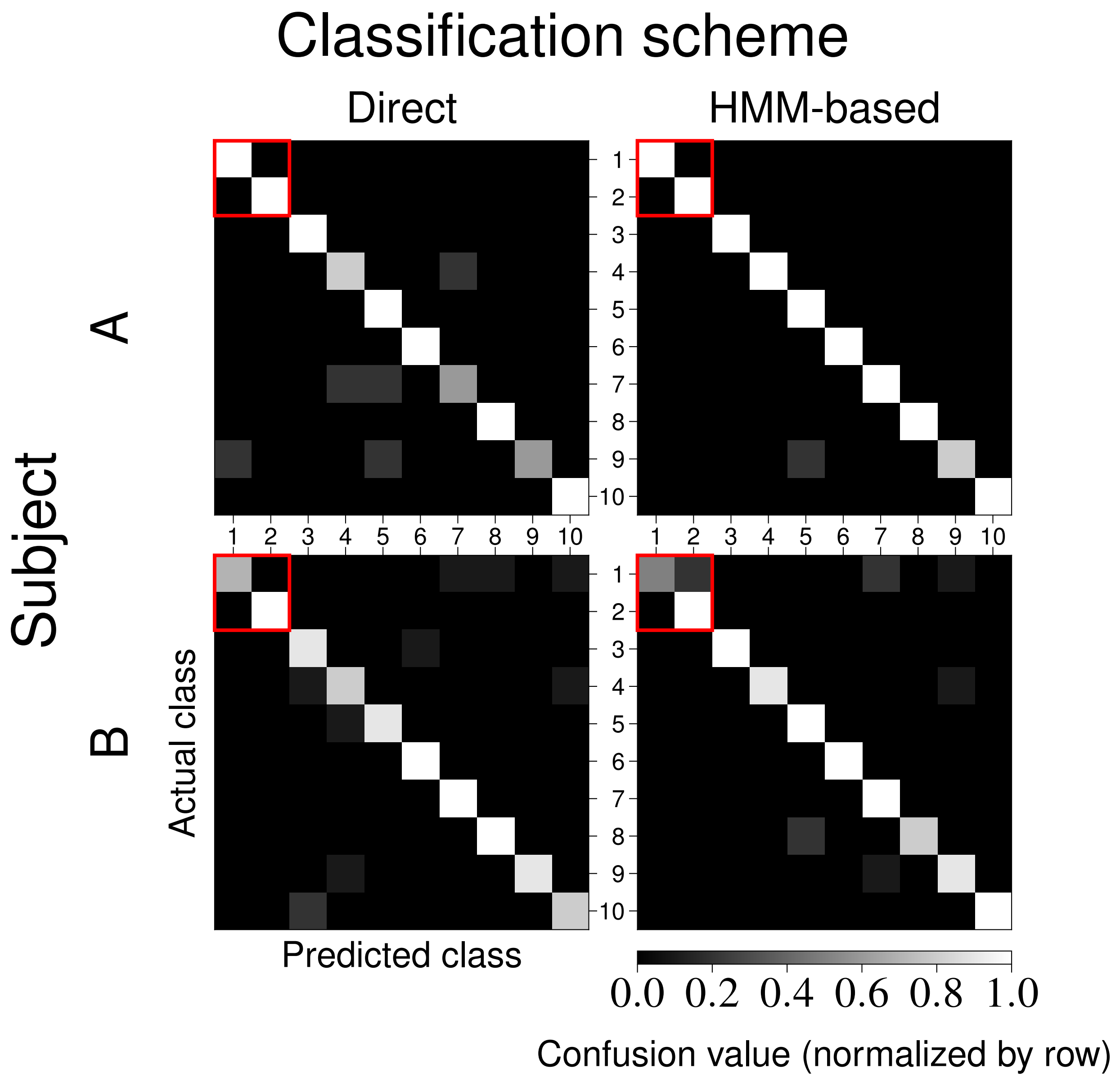


Figure 3.6: Confusion matrices computed using the final task block for each subject and classification scheme. Each row depicts results for a single subject (A or B) and each column depicts results for a single classification scheme (direct or HMM-based). The class numbers 1-10 correspond to the stimulus order given in Table 3.1. The color-value mapping is identical across all confusion matrices and uses row-normalized confusion values. The red outline signifies the two stimuli (classes 1 and 2) that were produced by the same speaker (all other stimuli were produced by unique speakers). In general, these two stimuli were not confused with each other, suggesting that the classifiers were not relying on speaker identity to make predictions. During HMM-based classification with subject B, class 1 was confused with class 2 20% of the time, but it was also confused with classes 7 and 9 20% and 10% of the time, respectively.

Chapter 4

Real-time decoding of question-and-answer speech dialogue using human cortical activity

Disclaimer: This chapter contains material from work that is currently unpublished. The contents of this chapter are a direct adaptation of a manuscript that is currently being prepared for submission to a scientific journal. Before reading this chapter, I recommend searching for the related publication if it is available at that time. The publication may contain revised and/or additional results that are not available in this chapter. If the publication is available, its title should be very similar to the title of this chapter, I should be the first author, and the year of publication is expected to be 2018 or 2019.

Personal contributions: I developed real-time decoding system, performed all data collection and analyses, and wrote the current draft of the manuscript.

4.1 Abstract

The development of an advanced speech prosthesis relies on real-time decoding of speech from high-resolution neural signals. Previous work has demonstrated that it is possible to decode perceived or produced speech with some success in relatively constrained contexts. However, to our knowledge, no work has utilized a naturalistic task where perceived and produced speech are integrated, which could have practical applications for patients who are unable to communicate. Here, we demonstrate real-time decoding of perceived and produced speech from high-density electrocorticography (ECoG) activity in humans using a custom real-time neural speech recognition (*rtNSR*) software package. In our task, three human subjects each listened to questions (e.g., "When would you like me to check back on you?") and verbally produced answers (e.g., "Tomorrow"). The *rtNSR* system used high gamma activity (70–150 Hz) extracted from the ECoG signals to reliably detect when subjects were listening or speaking and then performed phone-level Viterbi decoding to predict the identity of each speech utterance. We leveraged the fact that certain answers were only plausible responses to certain questions to dynamically update the prior probabilities of each answer using the preceding question likelihoods predicted from ECoG activity. Our system was able to reliably decode speech utterances for each subject, with accuracy rates as high as 61% for produced answers and 75% for perceived questions (chance rates were 7% and 20%, respectively). Furthermore, integration of the decoded question likelihoods as context significantly improved answer decoding. We also show that high accuracy rates are achievable using only 15–20 minutes of training data, suggesting that this paradigm can be used practically in limited data settings. These results demonstrate that neural activity in speech perception and production regions can be used for real-time decoding of speech in natural, conversational settings.

4.2 Introduction

Multiple regions of the human cortex have been implicated in processing of perceived and produced speech (Boatman et al., 1997; Binder et al., 2000; Canolty et al., 2007; Mesgarani et al., 2014; Lotte et al., 2015; Carey et al., 2017; Conant et al., 2018; Chartier et al., 2018). Previous studies have successfully decoded acoustic (Pasley et al., 2012; Yang et al., 2015; Leonard et al., 2016) and phonemic (Mesgarani et al., 2014; Moses et al., 2016; Moses et al., 2018) representations of speech heard by human listeners directly from neural activity in the superior temporal gyrus (STG). Other work has characterized the functional role of the ventral sensorimotor cortex (vSMC) during speech production (Bouchard et al., 2013; Mugler et al., 2014; Lotte et al., 2015; Carey et al., 2017; Conant et al., 2018; Chartier et al., 2018). A few studies have demonstrated processing of speech-related neural activity from these regions in real-time, including the ability to map speech-evoked sensorimotor activations (Cheung and Chang, 2012), generate neural encoding models of perceived phonemes (Khalighinejad et al., 2017), decode produced isolated phonemes (Leuthardt et al., 2011), detect voice activity (Kanas et al., 2014), and classify perceived sentences (Moses et al., 2018). However, to the best of our knowledge no existing work has attempted to decode both perceived and produced speech simultaneously in real-time using an experimental task that mimics natural communication, which could have practical implications for the development of future assistive speech applications.

In this work, we demonstrate real-time decoding of perceived and produced speech from high-density electrocorticography (ECoG) activity in humans using a real-time neural speech recognition (*rtNSR*) software package that we developed (Moses et al., 2018). As described in our previous work, we use the term neural speech recognition to denote the decoding of speech using neural responses as features (Moses et al., 2016; Moses et al., 2018). In our task, human subjects first listen to a finite set of pre-recorded questions multiple times and verbally produce a finite set of answer responses multiple times while we collected time-aligned ECoG signals. After using this data for offline model fitting, we tested our *rtNSR* system by having the subjects listen to questions and respond aloud with answers while decoding the perceived questions and produced answers directly from neural activity in real-time (Fig. 4.1). From the neural signals, our system was able to detect when subjects were listening or speaking and predict the identity of each detected utterance using phone-level Viterbi decoding. Because certain answers were only valid responses to certain questions, we were able to integrate the question and answer predictions by dynamically updating the prior probabilities of each answer using the preceding predicted question likelihoods. We observed reliable decoding of both perceived and produced utterances in real-time, further validating our *rtNSR* system as a potential platform for speech neuroprostheses.

4.3 Results

4.3.1 Real-time decoding system overview

While human subjects performed a question-and-answer natural speech perception (Fig. 4.1a) and production (Fig. 4.1b) task, we recorded, processed, and analyzed concurrent neural activity from high-density ECoG arrays that covered auditory and sensorimotor cortical regions. In real-time, neural activity was filtered to extract signals in the high gamma frequency range (70–150 Hz; Fig. 4.1c, Fig. 4.7), which correlate with multiunit activity (Crone et al., 1998) and have been previously used to decode speech signals from auditory (Pasley et al., 2012; Mesgarani et al., 2014; Moses et al., 2016; Moses et al., 2018) and sensorimotor (Bouchard et al., 2013; Mugler et al., 2014; Lotte et al., 2015; Chartier et al., 2018; Conant et al., 2018) brain regions. We used these high-gamma signals to perform real-time speech event detection, predicting which time segments of the neural activity occurred during question perception (Fig. 4.1d, blue curve), answer production (Fig. 4.1d, red curve),

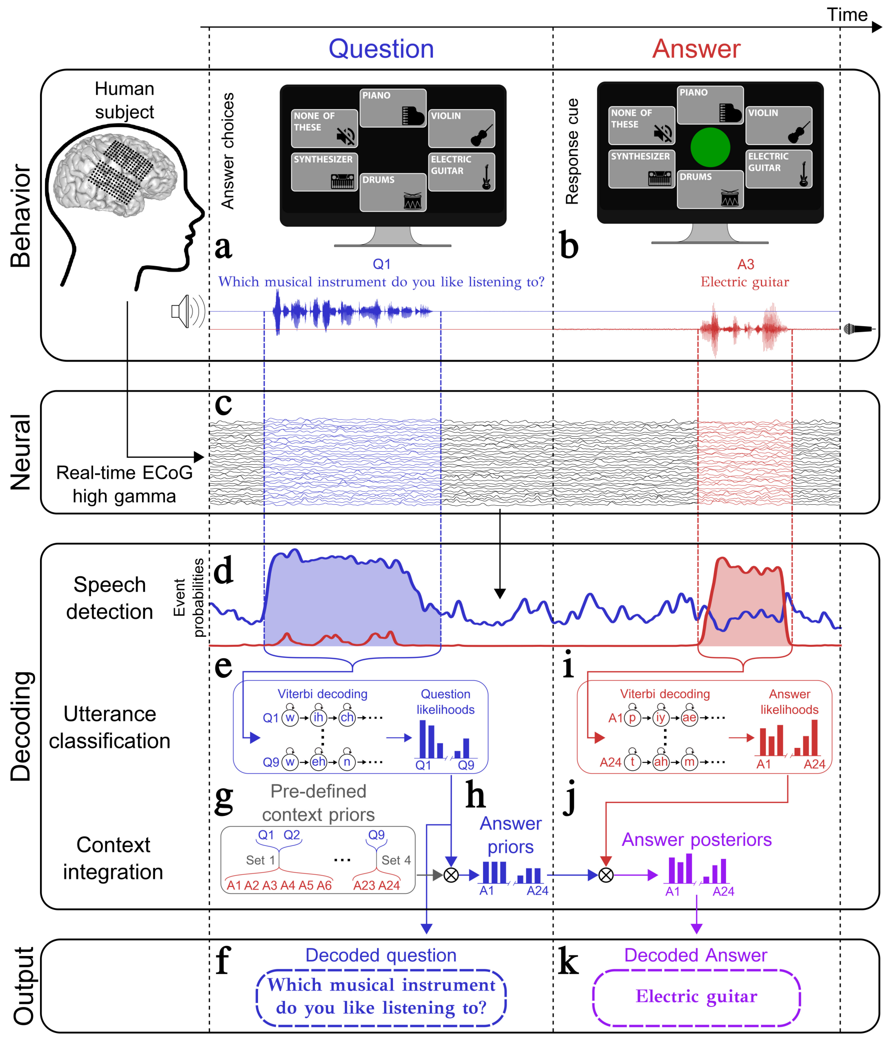


Figure 4.1: Schematic of real-time speech decoding during a question (blue) and answer (red) task. (**a**) For each trial, subjects hear a question, and a set of possible answer choices are presented on a computer monitor. (**b**) Subjects are instructed to choose and verbally produce one of the answers when a green response cue appears on the screen. (**c**) Cortical activity is acquired simultaneously from ECoG electrodes implanted across temporal and frontal cortex and is filtered in real-time to extract high gamma activity. (**d**) A speech detection model uses the high gamma signal across electrodes to estimate event probabilities that predict whether a question is being heard or an answer is being produced (or neither) at each time point. (**e**) When the speech detection model detects a question event, the associated time window of neural activity is passed to a question classifier that uses phone-level Viterbi decoding to compute question utterance likelihoods. (**f**) The question with the highest likelihood is output as the decoded question. (**g**) To integrate questions and answers, the stimulus set was designed such that each answer response was only valid for certain questions (context priors). (**h**) These context priors are combined with the predicted question likelihoods to obtain answer priors. (**i**) When the speech detection model detects an answer event, the associated time window of neural activity is passed to an answer classifier that uses phone-level Viterbi decoding to compute answer utterance likelihoods. (**j**) The context integration model combines these answer likelihoods with the answer priors to yield answer posterior probabilities (purple). (**k**) The answer with the highest posterior probability is output as the decoded answer.

or silence (curve not shown; see Fig. 4.9 and Section 4.5.6 for more details on the event detection procedure).

For each time segment associated with a detected question event, a question classification model processed the high gamma activity in this time segment to compute question likelihoods (Fig. 4.1e) using phone-level Viterbi decoding (Viterbi, 1967). In this approach, each question utterance was represented as a hidden Markov model (HMM) that modeled the probability of observing a time segment of high gamma activity assuming that the subject was hearing the sequence of phones that comprise the utterance. The most likely question was output as the decoded question (Fig. 4.1f).

To enhance answer decoding performance, we designed this question-and-answer task such that specific answer responses were only valid for certain questions (Table 4.3). For example, if a subject heard the question “How is your room currently?”, there were only five answers (“Bright”, “Dark”, “Hot”, “Cold”, and “Fine”) that he or she could respond with that would be reasonable given the context of this particular question. We used this relationship between each question and the valid answer responses for that question to define context priors (Fig. 4.1g). A context integration model combined these context priors with decoded question likelihoods to compute answer prior probabilities (Fig. 4.1h).

As with question decoding, for time segments associated with a detected answer event, an answer classification model processed the high gamma activity in this segment to compute answer likelihoods (Fig. 4.1i) using phone-level Viterbi decoding. The context integration model combined these answer likelihoods with the answer priors to obtain answer posterior probabilities (Fig. 4.1j), and the answer with the highest posterior probability was output as the decoded answer (Fig. 4.1k). The answer likelihoods (without context integration) were also stored for later offline comparisons.

Prior to testing, models were fit using data collected during separate training task blocks.

The question classification models were fit using data collected while subjects listened to multiple repetitions of each of the question stimuli they would hear in the testing blocks, and the answer classification models were fit using data collected while subjects read each answer response aloud multiple times (see Section 4.5.4). The speech detection models were fit using the available data from both of these types of training task blocks.

4.3.2 Question and answer decoding performance

In offline simulations of the real-time decoding procedure, we evaluated how well our system decoded questions, answers without context integration, and answers with context integration. The results for subject 1 are given in Fig. 4.2, and Fig. 4.5 contains the results for the other subjects. Precise significance testing statistics for all subjects are given in Table 4.1.

The primary performance evaluation metric was decoding accuracy rate for each test block, which was defined as 1 minus the utterance error rate using the actual and predicted utterances for each prediction type. Here, the utterance error rate in a test block was equal to the edit (Levenshtein) distance between the actual and predicted utterance sequences. Because the error rate measures the amount of deletions, insertions, and substitutions required to convert the predicted sequence of detected and classified utterances into the actual utterance sequence, we can use the decoding accuracy rate metric to describe the full performance of the system, incorporating contributions from the speech detection, utterance classification, and context integration models. For all subjects, accuracy rates for decoding of each prediction type (questions, answers without context, and answers with context) were significantly above chance ($P < 0.05$, one-tailed bootstrap test, P-values combined across test blocks using Fisher's method; Fig. 4.2a, Fig. 4.5a). Chance accuracy rate was computed using bootstrapped sequences of randomly-sampled utterances (see Section

4.5.9). Importantly, we also observed a significant increase in decoding accuracy rate during answer decoding when context was integrated compared to when it was not integrated ($P = 3.4 \times 10^{-3}$, $P = 3.4 \times 10^{-4}$, $P = 0.025$ for subjects 1–3, one-tailed permutation test, P-values combined across test blocks using Fisher's method). Overall, these accuracy rates for questions (2.9, 3.7, and 2.5 times the chance levels for subjects 1–3) and answers with context (8.8, 5.0, and 5.0 times the chance levels for subjects 1–3) were promising, demonstrating the efficacy of the various components of our decoding approach.

To evaluate the performance of the separate components of the full system, we also measured the performance of the utterance classification and context integration models (separately from the speech detection model) by performing predictions using the true event times determined from the acoustic transcriptions of the test blocks. For each subject, we calculated question and answer classification accuracy (the fraction of utterance classifications that were correct) using each trial across the test blocks for that subject. For all subjects, classification accuracies were above chance for each prediction type ($P < 0.05$, one-tailed bootstrapped Welch's t-test; Fig. 4.2b, Fig. 4.5b). As with the decoding accuracy rates, answer classification accuracies were higher when integrating context ($P = 0.033$, $P = 1.9 \times 10^{-6}$, $P = 9.2 \times 10^{-4}$ for subjects 1–3, one-tailed exact McNemar's test).

We also assessed classification performance using cross entropy, a metric that compares the predicted utterance likelihoods and the actual utterance identities for each trial across all test blocks for a subject. Here, cross entropy measures the average number of bits required to correctly identify the utterance label for a trial given utterance log likelihoods predicted by a classification model. These values provide further insight into the performance of the utterance classification and context integration models by considering the predicted probabilities of the utterances (not just which utterance was most likely in each trial). Lower cross entropy indicates better performance. For all subjects, cross entropies were below chance ($P < 0.05$, one-tailed bootstrap test) and were significantly lower for the answer predictions when integrating context ($P = 7.6 \times 10^{-6}$, $P = 2.6 \times 10^{-17}$, $P = 3.1 \times 10^{-11}$ for subjects 1–3, one-tailed Wilcoxon signed-rank test; Fig. 4.2c, Fig. 4.5c).

To evaluate the performance of the event detector, we computed a detection score that incorporates frame-by-frame detection accuracy and a comparison between the number of detected and actual utterances (Fig. 4.2d, Fig. 4.5d). For all subjects, detection scores for questions and answers were high (above 85%), consistent with the similar decoding accuracy rates and classification accuracies observed during testing.

Finally, to understand which brain regions contributed to the performance of each of these models, we calculated the discriminative power of each ECoG electrode (see Section 4.5.8). We found that for question decoding, discriminative power was highest primarily in STG (Fig. 4.2e, Fig. 4.5e), consistent with auditory responses to heard speech observed in this region. The electrodes that contributed most to answer decoding were located in both the vSMC and STG (Fig. 4.2f, Fig. 4.5f), reflecting responses both to speech production and perception of self-produced speech. Lastly, electrodes that contributed to speech detection were distributed throughout sensorimotor and auditory regions (Fig. 4.2g, Fig. 4.5g).

4.3.3 Classifier sensitivity to data limitations, hyperparameter selection, and cortical coverage

The results reported in this work depended on the particular circumstances of the three patients who participated in this study. To understand the limitations of the utterance classification models used in this task, we assessed their performance as a function of the amount of training data, the specific hyperparameters used during training and testing, and the functional-anatomical coverage of the ECoG electrodes. The results of these analyses for subject 3 are given in Fig. 4.3, and Fig. 4.6 contains the results for the other subjects.

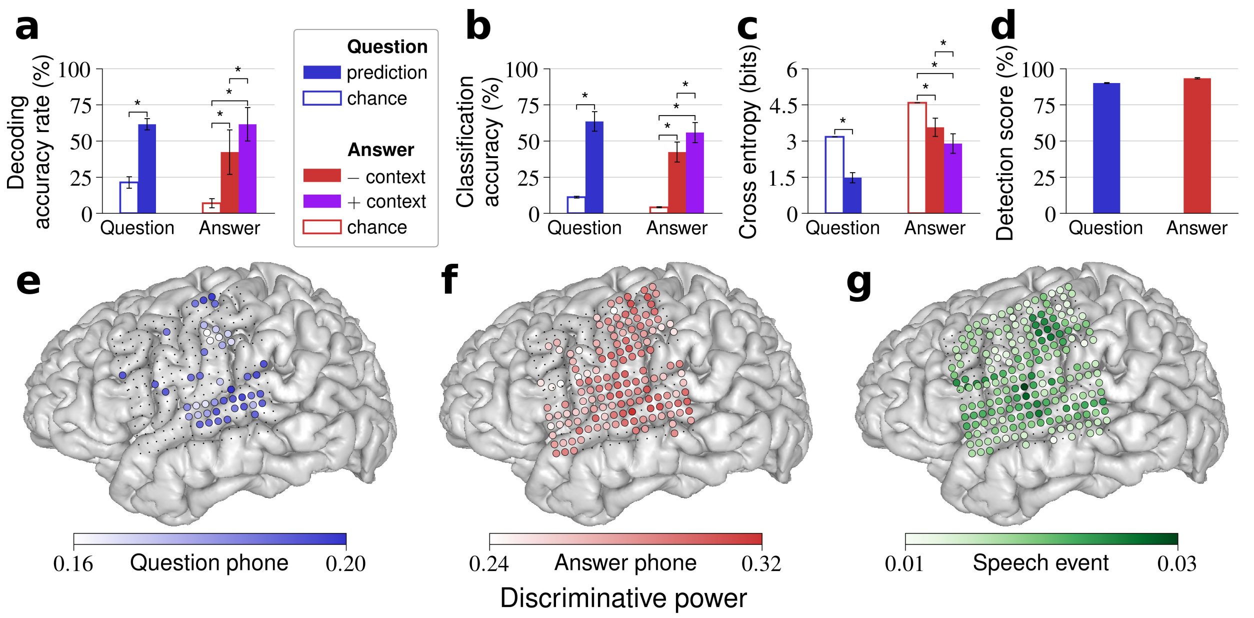


Figure 4.2: Decoding and classification results for questions, answers, and answers after integration of the decoded context for a single subject. (**a**) Decoding accuracy rates, which measure the full performance of the system, are significantly above chance for questions and answers (with and without context). Accuracy is significantly higher with context compared to without context. (**b**) Classification accuracies (the percent of speech events in which the system correctly classified the utterance) mirror decoding accuracy rates. (**c**) Cross entropies for utterance classification exhibit similar significant differences (lower values indicate better performance). (**d**) Question and answer detection scores demonstrate near-ceiling performance of the speech detection model for both questions and answers. (**e**–**g**) MRI brain reconstructions with electrode locations and discriminative power for each electrode used by **e** question phone, **f** answer phone, and **g** speech event discriminative models. Electrodes that were not relevant for the current model are depicted as small black dots. In **a**–**d**, data are mean ± SEM. $^*P < 0.05$.

First, we analyzed how the amount of neural data used during training affected classification performance. For each subject, we fit utterance classification models with neural data recorded during perception and production of a certain number of samples of each utterance (drawn randomly from the available training data). We then evaluated these models on all of the trials in the test blocks for that subject. In general, we found that classification accuracy and cross entropy improved over the first 10–11 training samples (Fig. 4.3a, Fig. 4.6a). Beyond 11 samples, performance began to improve more slowly, although performance never completely plateaued (except for the answer classifier for subject 2, Fig. 4.6a). These findings suggest that only 5–10 training samples of each utterance are required

Table 4.1: Significance testing statistics for question and answer decoding performance.

Metric	Test	Subject	Prediction type	P-value	Number of samples	Other information
Decoding accuracy rate	One-tailed bootstrap test, performance vs. chance, combined across blocks with Fisher's method	1	Question	≈ 0	2 (test blocks)	
			Answer without context	≈ 0		
			Answer with context	≈ 0		
		2	Question	≈ 0	4 (test blocks)	
			Answer without context	8.3×10^{-12}		
			Answer with context	≈ 0		
		3	Question	≈ 0	3 (test blocks)	
			Answer without context	≈ 0		
			Answer with context	≈ 0		
	One-tailed permutation test, with vs. without context, combined across blocks with Fisher's method	1	Answer (with vs. without context)	3.4×10^{-3}	2 (test blocks)	
		2	Answer (with vs. without context)	3.4×10^{-4}	4 (test blocks)	
		3	Answer (with vs. without context)	0.025	3 (test blocks)	
Classification accuracy	One-tailed bootstrapped Welch's t-test, performance vs. chance	1	Question	1.7×10^{-10}	52 (trials)	$t = 7.8, df = 51$
			Answer without context	5.9×10^{-7}		$t = 5.5, df = 51$
			Answer with context	5.9×10^{-10}		$t = 7.4, df = 51$
		2	Question	≈ 0	101 (trials)	$t = 23, df = 100$
			Answer without context	4.1×10^{-7}		$t = 5.3, df = 100$
			Answer with context	9.7×10^{-14}		$t = 8.5, df = 100$
		3	Question	4.0×10^{-14}	75 (trials)	$t = 9.2, df = 74$
			Answer without context	2.3×10^{-8}		$t = 6.1, df = 74$
			Answer with context	2.7×10^{-13}		$t = 8.7, df = 74$
	One-tailed exact McNemar's test, with vs. without context	1	Answer (with vs. without context)	0.033	52 (trials)	
		2	Answer (with vs. without context)	1.9×10^{-6}	101 (trials)	
		3	Answer (with vs. without context)	9.2×10^{-4}	75 (trials)	
Cross entropy	One-tailed bootstrap test, performance vs. chance	1	Question	5.6×10^{-16}	52 (trials)	
			Answer without context	3.3×10^{-3}		
			Answer with context	1.3×10^{-5}		
		2	Question	≈ 0	101 (trials)	
			Answer without context	1.0×10^{-5}		
			Answer with context	≈ 0		
		3	Question	≈ 0	75 (trials)	
			Answer without context	3.7×10^{-11}		
			Answer with context	≈ 0		
	One-tailed Wilcoxon signed-rank test, with vs. without context	1	Answer (with vs. without context)	7.6×10^{-6}	52 (trials)	
		2	Answer (with vs. without context)	2.6×10^{-17}	101 (trials)	
		3	Answer (with vs. without context)	3.1×10^{-11}	75 (trials)	

to achieve high performance, but it remains unclear how many training samples would be required before performance no longer improves.

Next, we investigated the impact that hyperparameter selection had on classification performance. The classifiers we used in this work contained parameters that were not learned directly from the training data (such as the parameters that controlled how the neural data was structured into feature vectors during training and testing). Due to clinical time constraints, we were unable to methodically set these values before online testing at the patient's bedside. Instead, prior to evaluating the performance of our system with real-time offline simulations, we performed cross-validated hyperparameter optimization on the models used during decoding (see Section 4.5.7). Using an iterative optimization algorithm, we evaluated 250 different sets of hyperparameter values for each test block using a leave-one-block-out cross-validation scheme. The hyperparameter values used during the primary performance evaluations for a test block were the values that exhibited the best performance on the held-out validation set for that block. To better understand how hyperparameter selection affected performance, we obtained all 250 of the hyperparameter sets that were considered during optimization for one of the test blocks for each subject, and we then evaluated these hyperparameter values on that test block itself (instead of on the validation set). For each subject, we observed large variabilities in classification accuracy and cross entropy across the different hyperparameter sets, suggesting that hyperparameter values can have a large impact on performance (Fig. 4.3b, Fig. 4.6b). For each subject and metric, we also found that the hyperparameters that were deemed optimal on the validation set were always better than the median performance observed across all hyperparameter sets, demonstrating that the optimizer was able to choose reasonable hyperparameter values to use during testing.

Finally, we assessed the effect that limiting cortical coverage had on the classification models. We separated the electrodes for each subject into two functional-anatomical subsets using the Sylvian fissure as the dividing line (Fig. 4.3c, Fig. 4.6c). Neural activity recorded from the infra-Sylvian electrodes, which sampled from cortical regions associated with auditory processing (including the STG), were used to fit question classification models.

Neural activity from the supra-Sylvian electrodes, which sampled from cortical regions associated with speech motor processing (including the sensorimotor cortex), were used to fit answer classification models. The context integration model combined question likelihoods from the infra-Sylvian model and answer likelihoods from the supra-Sylvian model to predict answers with context integration. For each subject, we evaluated these models on all test blocks using the classification accuracy and cross entropy metrics. Classification accuracies were above chance for all subjects and prediction types ($P < 0.05$, one-tailed bootstrapped Welch's t-test; Fig. 4.3d, Fig. 4.6d; precise significance testing statistics given in Table 4.2) and were improved for answer predictions when integrating context for subjects 2 and 3 but not for subject 1 ($P = 0.25390625$, $P = 3.8 \times 10^{-6}$, $P = 2.4 \times 10^{-4}$ for subjects 1–3, one-tailed exact McNemar's test). Cross entropies were below chance for all subjects and prediction types except for the answer predictions without context for subject 1 ($P = 1.2 \times 10^{-11}$, $P = 0.065$, $P = 1.2 \times 10^{-5}$ for questions, answers without context, and answers with context for subject 1, $P < 0.03$ for subject 2, $P < 10^{-6}$ for subject 3, one-tailed bootstrap test). For all subjects, cross entropies were lower for the answer predictions when integrating context ($P = 1.4 \times 10^{-8}$, $P = 1.3 \times 10^{-18}$, $P = 1.3 \times 10^{-7}$ for subjects 1–3, one-tailed Wilcoxon signed-rank test). Overall, these results demonstrate that utterance classification is possible when only auditory or only sensorimotor cortical areas are used, but having access to both regions improves performance.

4.3.4 Viterbi classification and phonetic modeling

To gain a more intuitive understanding of the neural and stimulus-dependent features used by the decoder, we examined the specific phonetic decisions the model made during answer classification (Fig. 4.4). The classifiers were trained by fitting phone likelihood estimation models using neural data collected during training blocks along with time-aligned phonetic transcriptions of the acoustics (see Section 4.5.6). These phone models computed likelihoods

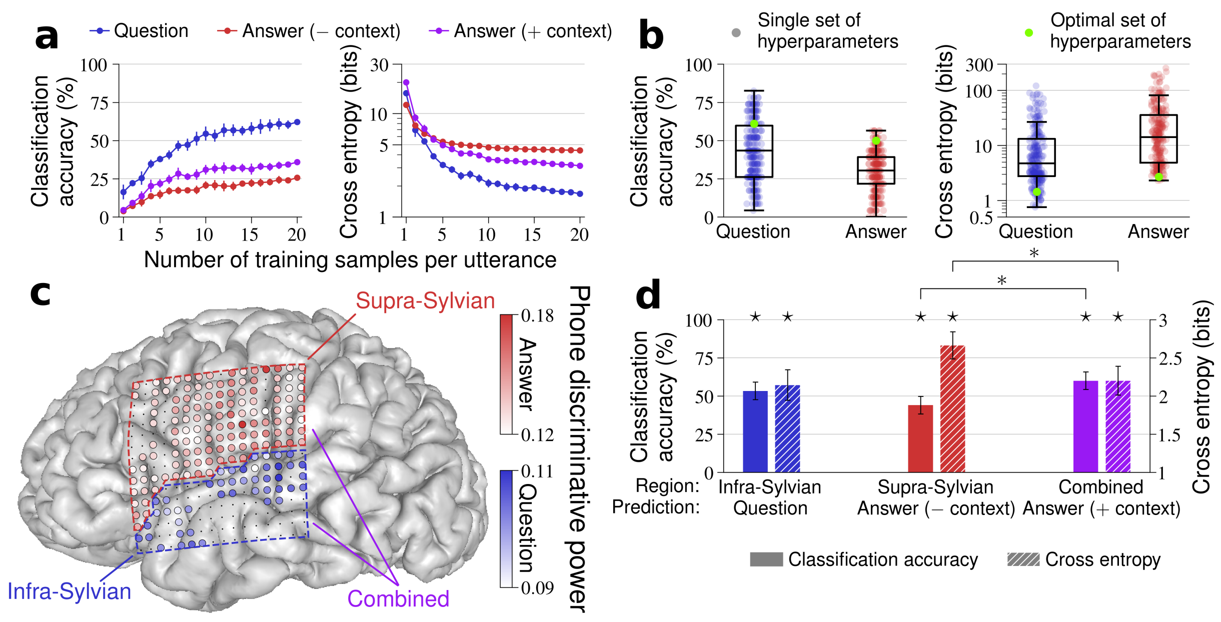


Figure 4.3: Data limitations, hyperparameter optimization, and functional anatomy of speech classification for a single subject. (**a**) Classification accuracy and cross entropy as a function of the amount of training data. The performance improvements appear to slow down slightly once 10–11 samples of each utterance are used, but performance does not seem to ever plateau completely for this subject. (**b**) Variability in classification performance across hyperparameter optimization epochs for one test block. Each blue and red dot shows the performance on the test block using a single set of hyperparameters chosen for one epoch during optimization on a separate validation set. Each boxplot depicts a line marking the median value, box heights representing the interquartile range, and whiskers extending beyond the box edges by 1.5 times the interquartile range. Each green dot marks the performance on the test block using the hyperparameters that minimized cross entropy on the validation set. This demonstrates that hyperparameter selection has a large impact on performance and that the optimizer is able to choose hyperparameter values effectively. (**c**,**d**) Contributions of distinct brain regions on classification performance. **c** MRI reconstruction with electrode locations, separated by the Sylvian fissure into infra-Sylvian (auditory) and supra-Sylvian (motor) cortical regions. In this analysis, we fit question classification models using only infra-Sylvian electrodes and answer classification models using only supra-Sylvian electrodes. Answer predictions with context were computed using a combined model that integrated the question likelihoods from the infra-Sylvian model with the answer likelihoods from the supra-Sylvian model. The phone discriminative powers estimated for each relevant electrode are shown as large colored circles, and the electrodes that were not relevant are depicted as small black dots. **d** Classification performance using these region-of-interest classification models. Here, all models performed above chance ($^{\star}P < 0.05$), and performance was significantly improved when integrating context using the combined model ($^{*}P < 0.05$). These results demonstrate that reliable performance can be achieved from the classification models in scenarios involving limited cortical coverage from a particular subject. In (**a**,**d**), data are mean ± SEM.

for each phone at individual time points using neural features. The classifiers represented each utterance as a hidden Markov model (HMM), with phones as hidden states and neural

Table 4.2: Significance testing statistics for the region-limited analysis.

Metric	Test	Subject	Prediction type	P-value	Number of samples	Other information
Classification accuracy	One-tailed bootstrapped Welch's t-test, performance vs. chance	1	Question (infra-Sylvian)	4.1×10^{-13}	52 (trials)	$t = 9.5, df = 51$
			Answer (supra-Sylvian)	7.1×10^{-5}		$t = 4.1, df = 51$
			Answer (combined)	7.1×10^{-6}		$t = 4.8, df = 51$
		2	Question (infra-Sylvian)	≈ 0	101 (trials)	$t = 19, df = 100$
			Answer (supra-Sylvian)	5.4×10^{-3}		$t = 2.6, df = 100$
			Answer (combined)	4.8×10^{-8}		$t = 5.8, df = 100$
		3	Question (infra-Sylvian)	1.5×10^{-10}	75 (trials)	$t = 7.3, df = 74$
			Answer (supra-Sylvian)	7.4×10^{-10}		$t = 6.9, df = 74$
			Answer (combined)	2.6×10^{-15}		$t = 9.8, df = 74$
	One-tailed exact McNemar's test, with vs. without context	1	Answer (combined vs. supra-Sylvian only)	0.25	52 (trials)	
		2	Answer (combined vs. supra-Sylvian only)	3.8×10^{-6}	101 (trials)	
		3	Answer (combined vs. supra-Sylvian only)	2.4×10^{-4}	75 (trials)	
Cross entropy	One-tailed bootstrap test, performance vs. chance	1	Question (infra-Sylvian)	1.2×10^{-11}	52 (trials)	
			Answer (supra-Sylvian)	0.065		
			Answer (combined)	1.2×10^{-5}		
		2	Question (infra-Sylvian)	≈ 0	101 (trials)	
			Answer (supra-Sylvian)	0.021		
			Answer (combined)	≈ 0		
		3	Question (infra-Sylvian)	1.1×10^{-7}	75 (trials)	
			Answer (supra-Sylvian)	≈ 0		
			Answer (combined)	≈ 0		
	One-tailed Wilcoxon signed-rank test, with vs. without context	1	Answer (with vs. supra-Sylvian only)	1.4×10^{-8}	52 (trials)	
		2	Answer (with vs. supra-Sylvian only)	1.3×10^{-18}	101 (trials)	
		3	Answer (with vs. supra-Sylvian only)	1.3×10^{-7}	75 (trials)	

data as observed states. During testing, we performed Viterbi decoding on the HMM associated with each utterance to compute the most likely path through the hidden states (phones) given the observed sequence of neural data. The resulting log probabilities of these Viterbi paths from each HMM were used to compute the likelihood of each utterance.

With these utterance classifiers, we examined how phone likelihood estimates affected the probability of each utterance across time. For example, when a subject produced the answer "Fine" (in response to the question "How is your room currently?"), an answer classifier used the sequence of phone likelihood estimates (predicted from neural data) to update the predicted probabilities of each possible answer at each time point during the utterance (Fig. 4.4a). The pattern of utterance probabilities illustrates how utterances with phonetic content

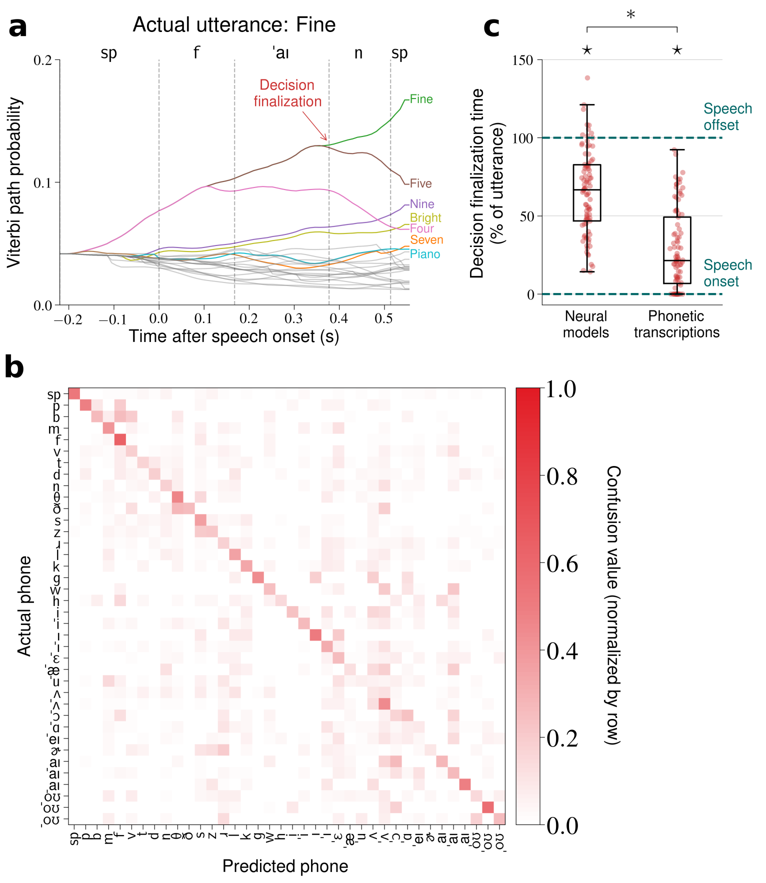


Figure 4.4: Temporal characteristics and phone-based performance of the answer (speech production) classification model. (**a**) Example Viterbi path probabilities during production of the utterance “Fine” during a test block demonstrate how the classifier uses phone-level information to predict answers as speech unfolds over time. Each curve tracks the probability of each answer utterance as the classifier receives additional neural data. The probability values at the final time point represent the answer likelihoods that are passed to the context integration model. Only the seven most likely utterances are labeled and colored for visualization purposes. The time at which the correct utterance becomes more likely than the other utterances (and remains more likely throughout the remainder of the decoding window) is marked as the “Decision finalization” time. (**b**) Phone confusion matrix using the answer phone likelihood model for every time point in each test block across all subjects. This matrix demonstrates reliable discrimination between the majority of the phones and intuitive confusions within articulatory classes (e.g., /s/ vs. /z/). (**c**) Decision finalization times for the answer classifications using neural data and the phonetic transcriptions across all subjects and test blocks. Each red dot represents the decision finalization time for a correctly-predicted trial (plotted as percent of the length of the utterance relative to the actual speech onset and offset for that trial). Each boxplot depicts a line marking the median value, box heights representing the interquartile range, and whiskers extending beyond the box edges by 1.5 times the interquartile range. The observed finalization times occurred before speech offset, indicating that the classifiers were able to predict the identity of an utterance before processing the latter time points in the neural (or phonetic) time window associated with an utterance ($^{\star}P < 10^{-14}$). This characteristic is only partially explained by the stimuli and transcribed vocalizations ($^{*}P < 10^{-9}$).

similar to that of the actual utterance were assigned higher likelihoods during time segments of phonetic overlap. For example, the utterances “Fine” and “Five” remain equally likely until the neural activity collected during production of the /n/ phone is seen (at which time “Fine” becomes the most likely utterance).

Across all utterances and all phones, we found that the answer phone likelihood models were able to reliably classify most of the phones (Fig. 4.4b). Where there were phone confusions, these confusions typically clustered according to interpretable articulatory classes. For example, /t/ and /d/ were confused because they are both alveolar stops that differ only in voice onset time.

Finally, to understand how much phonetic information the answer classifiers needed each trial before finalizing the utterance prediction, we assessed the earliest time point during the Viterbi decodings for each HMM at which the utterance that ends up being most likely at the end of decodings becomes and remains more likely than the other utterances. We defined the decision finalization time as the fraction of the utterance that had been uttered when this time point was reached (using the actual speech onset and offset times). We computed these decision finalization times for each trial in which the answer classification models correctly predicted the answer response (94 trials total across all subjects and test blocks). We found that the classifiers were typically able to finalize their decisions before all of the neural data from an utterance was seen ($P = 2.1 \times 10^{-15}$, one-tailed single-sample Wilcoxon signed-rank test; Fig. 4.4c). Because some utterances begin with the same phones (e.g., the phones /s ˈɪ/ at the start of “Six” and “Synthesizer”), we expected the lower bound for the finalization times to occur after speech onset even if the actual phone identity at each time point were known. To compute this lower bound, we re-calculated the finalization times for these trials using phone likelihoods constructed directly from the phonetic transcriptions. Because no two utterances had the same phonetic content, these transcription-based finalization times always occurred before the speech offset ($P = 1.6 \times 10^{-16}$, one-tailed single-sample Wilcoxon signed-rank test). The neural-based finalization times were significantly higher than the transcription-based finalization times ($P = 1.2 \times 10^{-10}$, one-tailed Wilcoxon signed-rank test), which can be attributed to imperfect phone likelihood estimation (if phone likelihood estimation were perfect, the neural-based and transcription-based finalization times would be identical). Overall, these results demonstrate that the answer classifiers were able to finalize classification decisions before the offset of speech using estimated phone likelihoods. Furthermore, this observation cannot be explained entirely by the phonetic separability of the utterances themselves.

4.4 Discussion

Overall, these results demonstrate that perceived and produced speech can be reliably decoded from human cortical activity in real-time using a naturalistic question-and-answer task paradigm. For each subject, we showed that the high gamma activity collected during the task can be used to detect when a subject was hearing or producing speech, predict what a subject heard or said, and improve the predictions of what was said using contextual information inferred from the predictions of what was heard. We observed decoding accuracy rates as high as 61% for produced speech and 75% for perceived speech, which were both significantly above chance levels. These findings represent an important step towards development of advanced speech neuroprostheses, exhibiting for the first time that volitionally-chosen spoken speech responses can be decoded in real-time from neural signals after integration of the predicted contextual state of the speaker.

Although the ability to detect produced speech from neural activity has been previously demonstrated (Kanas et al., 2014), the speech detector used in this work is the first to demonstrate reliable discrimination between perceived speech, produced speech, and silence in real-time from neural signals. Our speech detector was relatively simple, relying on speech event probabilities from linear models of neural activity, but also flexible, using hyperparameters that could be set via optimization to determine how to use the predicted speech event probabilities to determine speech onsets and offsets. Instead of providing the model with task-relevant information to improve performance (for example, we could have forced the detector to expect an answer event after detecting a question event), we decided to keep the detector more generalized, allowing it to detect questions and answers independently of previous detections. Because of its general and flexible implementation, the detector should be able to perform similarly in tasks involving production of continuous, large-vocabulary, and natural speech sentences. Future research could also investigate how well this approach could be applied towards detection of imagined speech events, which would have major implications for speech neuroprosthetic applications.

Our utterance classification models were successfully able to model the relationship between neural activity and phones and leverage that ability to discriminate between utterances. By using HMMs to represent the utterances and Viterbi decoding to compute utterance likelihoods, both the question and answer classifiers were robust to slight inaccuracies in the detected speech onsets and offsets, and the answer classifiers were robust to variability in the exact pronunciations of the answer responses. Although previous studies have provided significant evidence supporting the encoding of phonemes in STG activity during speech perception (Mesgarani et al., 2014; Leonard et al., 2016), other studies suggest that articulator kinematics (e.g., lip and tongue movements) explain more variance in vSMC activity than phonemes (Chartier et al., 2018). This suggests that, despite reliable phone discrimination, our answer phone likelihood models were merely learning the phonetic correlations with the true underlying articulatory representations, meaning that more powerful answer classifiers could be designed that rely on prediction of articulator kinematics as opposed to phones.

For all subjects, we observed that predicted question likelihoods could be used as context to improve subsequent answer predictions. When using the question likelihoods as context, it was only important that the questions in the correct question/answer (QA) set received a large total amount of the predicted probability mass, not that the correct question itself had a high likelihood value. Because of this, we designed the question stimuli such that the initial part of each question utterance within a QA set was identical to the other questions in that set but different from questions in the other QA sets. We also hypothesized that we would observe accurate question decoding based on the well-established phonemic representations in the STG (Mesgarani et al., 2014), our previous work involving classification and decoding of perceived phonemes (Moses et al., 2016; Moses et al., 2018), and the small number of question utterances used in this task. These factors were all considered when deciding to use question decodings as contextual information and all most likely contributed to the success of the question classification and context integration models. Although there were few contextual states (QA sets) and target outputs (answer utterances), this context integration approach could be extended to incorporate various input sources as context for large-vocabulary decoding. For example, in an assistive speech application for a patient with limited communicative capabilities, it could be possible to apply standard automatic speech recognition techniques to decode questions directly from acoustics (which should be more accurate than decoding from neural activity). Additionally, information inferred from camera inputs (e.g., objects or people near the patient), location inputs (e.g., whether the patient is at home or in a hospital), or direct computer inputs (e.g., the program or website that the patient is trying to interact with) could all be used as context to impose situationally-relevant priors during decoding of intended speech or actions.

The design of the task used in this work was largely dependent on our goals for decoding multiple speech modalities and assessing context integration efficacy. We also wanted our task to mimic natural conversation by having subjects decide which answer to respond with in each trial. Even though the choices were limited, subjects were not simply reading from prompts during testing, suggesting that our approach is viable for more complicated spontaneous speech tasks. The stimuli used in the task were designed with the decoding and context integration goals in mind, but were also qualitatively chosen to be colloquial and clinically relevant. Overall, subjects seemed engaged and responsive during data collection, with less than 0.5% of answer trials containing no utterance or an incorrect utterance.

All of our testing was dependent on the clinical constraints of the patients who participated in this study. These constraints, including clinical schedules, ECoG grid placement, fatigue, and medication, affected the amount of data we could collect during each experimental session, the number of sessions we could have with each subject, and the cortical regions that we could use for decoding. Even though we were able to get reliable classifications after about 10 samples of each utterance, our findings suggested that performance could have been further improved if we were able to collect more data from each subject. Similarly, the number of sessions that we had with each subject prevented us from using optimized models during online testing, which, based on our findings, would have led to improved decoding during these tests. Although we cannot definitively describe how the cortical coverage that we had with each subject affected performance, we typically observed better accuracy when having access to all of the electrodes instead of being limited to electrodes from functionally-relevant regions. In future applications of this approach in dedicated settings, with the ability to control ECoG grid placement and collect data more liberally over multiple days before optimization and testing, we are confident that these techniques would exhibit even better speech decoding performance.

In future work, we intend to continue augmenting the capabilities of the *rtNSR* system to further improve performance and expand the range of testing paradigms that it is compatible with. For example, future iterations of the system could include automated bad channel and time segment rejection, which would help to maintain the quality of the signals that are used during model training. Although abnormal activity (caused by epileptic spiking, faulty electrode contacts, or other factors) might have been present in the signals we collected for this work, it would be more likely to negatively affect performance than to improve it, meaning that reliable rejection of these abnormalities should lead to increases in decoding accuracy. Additionally, the HMM-based classifiers could be replaced by recurrent neural network classifiers, which should be able to learn the temporal structure of the neural representations of speech without being limited to linear modeling or explicit assumptions about the underlying representations (given sufficient training data). We can also explore feedback and subject adaptation methodologies, which would be useful in closed-loop experiments and practical applications.

Eventually, we hope to deploy the *rtNSR* system as part of an assistive speech application aimed at restoring communicative capabilities to patients that are unable to communicate. For some impaired individuals, such as patients with locked-in syndrome who are conscious but unable to communicate naturally due to paralysis (American Congress of Rehabilitation Medicine, 1995; Laureys et al., 2005; Bruno et al., 2011; Rousseau et al., 2015), restoration of limited communicative capability is associated with significant increases in self-reported quality of life (Bruno et al., 2011; Rousseau et al., 2015). Although the current state-of-the-art speech prostheses are already beneficial to patients, they are slow and unnatural, requiring patients to spell out intended messages slowly at rates less than 2 characters per minute (Sellers et al., 2014; Vansteensel et al., 2016). An ideal speech prosthesis would be capable of decoding spontaneous, natural speech controlled by a patient's volition. The results of this work are a promising step towards this goal, demonstrating that produced speech can be detected and decoded from neural activity in real-time while integrating dynamic information from other modalities.

4.5 Methods

4.5.1 Subjects

The three subjects who participated in this study were human epilepsy patients undergoing treatment at the UCSF Medical Center. For the clinical purpose of localizing seizure foci, ECoG arrays were surgically implanted on the cortical surface of each subject. All subjects were right-handed and determined to have left hemisphere language dominance by their clinicians.

The research protocol was approved by the UCSF Committee on Human Research. Prior to surgery, each patient gave his or her informed consent to be a subject for this research.

4.5.2 Neural data acquisition

Subjects 1 and 2 were each implanted with two 128-channel ECoG arrays (PMT corp.) and subject 3 was implanted with a 256-channel ECoG array (Ad-Tech, Corp.). Subjects 1 and 3 had left hemisphere coverage and subject 2 had right hemisphere coverage. Each implanted array contained disc electrodes with 1.17 mm exposure diameters arranged in a square lattice formation with a 4 mm center-to-center electrode spacing. We used the open source *img_pipe* package (Hamilton et al., 2017) to generate MRI brain reconstruction images with electrode locations for each subject (Fig. 4.2, Fig. 4.5).

We used a data acquisition (DAQ) rig to process the local field potentials recorded from these arrays at multiple cortical sites from each subject during experimentation. These analog ECoG signals were amplified and quantized using a pre-amplifier (PZ5, Tucker-Davis Technologies). We then performed anti-aliasing (low-pass filtering at 1500 Hz) and line noise removal (notch filtering at 60, 120, and 180 Hz) on a digital signal processor (RZ2, Tucker-

Davis Technologies). On the DAQ rig, we stored these neural data (at 3051.76 Hz) along with the time-aligned microphone and speaker audio channels (at 24414.06 Hz). These neural data were anti-aliased again (low-pass filtered at 190 Hz) and streamed at a sampling rate of 381.47 Hz to our real-time computer, which was a Linux machine implementing *rtNSR* (64-bit Ubuntu 14.04, Intel Core i7-4790K processor, 32 GB of RAM).

4.5.3 High gamma feature extraction

Within *rtNSR*, we implemented a filter chain comprised of three processes to measure high gamma activity in real-time (Fig. 4.7). We used high gamma band activity (70–150 Hz) in this work because previous research has shown that activity in this band is correlated with multi-unit firing processes in the cortex (Crone et al., 1998) and can be used as an effective representation of cortical activity during speech processing (Pasley et al., 2012; Bouchard et al., 2013; Mesgarani et al., 2014; Moses et al., 2016; Moses et al., 2018).

The first of these three processes applied eight band-pass finite impulse response (FIR) filters to the ECoG signals acquired from the DAQ rig (at 381.47 Hz). These filters were designed to approximate the offline high gamma band-pass filtering approach used in our lab's previous work (Bouchard et al., 2013; Moses et al., 2016). The logarithmically increasing center frequencies of these filters were 72.0, 79.5, 87.8, 96.9, 107.0, 118.1, 130.4, and 144.0 (in Hz, rounded to the nearest decimal place). The filters each had an order of 150 and were designed using the Parks-McClellan algorithm (Parks and McClellan, 1972).

The second process in the filter chain estimated the analytic amplitude values for each band and channel using the signals obtained from the band-passing process. An 80th-order FIR filter was designed using the Parks-McClellan algorithm to approximate the Hilbert transform. For each band and channel, this process estimated the analytic signal using the original signal (delayed by 40 samples, which was half of the filter order) as the real component and the FIR Hilbert transform approximation of the original signal as the imaginary component (Romero and Jovanovic, 2012). The analytic amplitudes were then computed as the magnitudes of these analytic signals. This filtering approach was only applied to every fourth sample of the received signals, effectively decimating the signals to 95.37 Hz.

The final process in the filter chain averaged analytic amplitude values across the eight bands, yielding a single high gamma analytic amplitude measure for each channel.

After filtering, the high gamma signals were then z-scored using a 30-second sliding window. For online, real-time speech decoding, it is not possible to do hand-labeled artifact rejection. Therefore, to mitigate signal artifacts and outliers caused by channel noise, epileptic activity, or other factors, the z-score values were clipped to lie within the range of $[-3.5, 3.5]$. We used the resulting z-scores as the representation of high gamma activity in all subsequent analyses and real-time testing.

4.5.4 Experimental task design

The overall goal of this task was to demonstrate real-time decoding of perceived and produced speech while leveraging predicted contextual information. To achieve this, we designed a question-and-answer task paradigm that involved a subject listening to question stimuli and responding verbally to each question with an answer. In total, we used 9 pre-recorded acoustic question stimuli and 24 visual answer stimulus prompts (Table 4.3). All question stimuli were recorded from the same female speaker at 44.1 kHz and were presented to the subject aurally via loudspeakers during the task blocks. Each visual answer stimulus was represented as a small rectangle containing the text prompt and a small image depicting the text (Fig. 4.1b; images were only included to improve subject engagement). The questions and answers were divided into four distinct question/answer sets (QA sets 1–4). The answers

Table 4.3: The question/answer sets.

QA set number	Question	Answer
1	Which musical instrument do you like listening to? Which musical instrument do you dislike hearing?	Piano Violin Electric guitar Drums Synthesizer None of these
2	How is your room currently?	Bright Dark Hot Cold Fine
3	From 0 to 10, how much pain are you in? From 0 to 10, how nauseous are you? From 0 to 10, how happy do you feel? From 0 to 10, how stressed are you? From 0 to 10, how comfortable are you?	Zero One Two Three Four Five Six Seven Eight Nine Ten
4	When do you want me to check back on you?	Today Tomorrow

in each QA set represented the valid answer choices for each of the questions in that set.

We used three different types of task blocks during experimentation: question (perception) training, answer (production) training, and testing task blocks in which subjects heard questions and responded verbally with answers. During each task block, time-aligned behavioral and neural data were collected and stored. The data collected during training blocks were used to fit the decoding models. The data collected during testing blocks were used in real-time to decode the perceived questions and produced answers and were also used offline during hyperparameter optimization.

Question (perception) training task block

In the question training block, subjects were presented with 10 repetitions of each question stimulus. Subjects were instructed to remain alert while listening to the questions without responding. Each question was between 1.38 and 2.42 seconds long and was presented with a constant onset-to-onset interval of 3 seconds.

Answer (production) training task block

Two different answer training blocks were used during training: one block containing answers from QA sets 1–2 and another block containing answers from QA sets 3–4. In each answer training block, subjects were presented with 10 repetitions of each answer choice. During each trial, one of the visual answer stimuli appeared on a screen in front of the subject with a gray background for 0.5 seconds. Afterward, the background of the stimulus changed to green for 1.5 seconds. Subjects were instructed to speak the text contained in the visual stimulus when the background of the stimulus changed to green. At the end of each trial, the screen was cleared and remained blank for 0.5 seconds before the start of the next trial.

Testing block

After collecting the training blocks and performing offline transcriptions and modeling, subjects participated in testing blocks to evaluate our real-time decoding system. At the start of each trial, a question was played to the subject while the valid answer stimuli from that QA set were presented visually to the subject in a circular outline arrangement on the screen. After 2.5–4.5 seconds from the start of the question, a green circle appeared in the middle of the screen (in the center of the presented answer choices) for 1.5–2 seconds. Subjects were instructed to say one of the answer choices when this green circle appeared. Subjects were encouraged (but not required) to choose answer choices freely without trying to answer the questions based on his or her current mood and to choose a variety of answer choices instead of saying the same answer choice in each trial. At the end of each trial, the screen was cleared and remained blank for 2–3 seconds before the start of the next trial.

The times given above were constant throughout any single testing block but were sometimes adjusted between blocks and between subjects depending on the subject's ability to choose and verbalize an answer choice in the allotted time. In each block, the questions played to the subject were chosen based on how many questions and answers are in each QA set (questions with more valid answers have a greater chance of being played in each trial). Any trial in which the subject failed to respond or responded with an invalid choice was excluded from further analysis. There were 26 question-and-answer trials in each testing block.

4.5.5 Phonetic transcription

After data collection, we phonetically transcribed the recorded acoustics using the *p2fa* package (Yuan and Liberman, 2008), which relies on the Hidden Markov Model Toolkit (HTK) and the CMU pronunciation dictionary (Young et al., 2002; Weide, 2014). The phone boundaries were manually fine-tuned within the *Praat* software package (Boersma, 2001). The phones were transcribed with ARPABET labels and then stored as International Phonetic Alphabet (IPA) labels. We included a silence phone token /sp/ to represent silence time points.

4.5.6 Modeling

After collecting training data for a subject, we fit models using the time-aligned high gamma z-score values and phonetic transcriptions. Model fitting was performed offline, and the trained models were saved to the real-time computer so that they could be used during online testing. As described later in Section 4.5.7, the values for many model parameters that were not learned directly from the training data were set using hyperparameter optimization. We used three types of models in this work: speech detection, utterance classification, and context integration models.

Speech detection

For each subject, we created speech detection models that processed neural activity and predicted which time segments occurred while the subject was perceiving a question or producing an answer. Each of these detection models analyzed the high gamma z-score values at every time point in real-time and determined the time points associated with the onsets and offsets of speech events. The primary goal of these detection models was to provide the utterance classification models (described in the next section) with time segments of neural data to use during prediction.

Before using the neural data to train speech detection models, we analyzed the collected data to identify which electrode channels should be considered relevant to detecting speech events (Moses et al., 2016; Moses et al., 2018). Using the phonetic transcriptions, we split the available high gamma z-score data into three subsets, each comprised of neural data that occurred during speech perception, speech production, or silence. We then performed Welch's analysis of variance (ANOVA) on each electrode channel individually to determine which channels were significantly modulated by the different types of speech events. Channels that had a Welch's ANOVA P-value less than a threshold hyperparameter were considered relevant and included in the feature vectors used during subsequent speech detection modeling.

A main component of each detection model was a speech event probability model capable of yielding $p\left(h_t|y_t\right)$ probabilities at every time point t during testing. Here, h_t represents the speech event at time t and is one of the values in the class set $\{perception, production, silence\}$, and y_t is the spatiotemporal neural feature vector at time t. The h_t labels were determined from the phonetic transcriptions: for any given time index t, h_t was *production* if the subject was producing a phone at time t, *perception* if the subject was listening to a phone at time t, or *silence* otherwise. Similar to the features used during modeling in our previous works (Moses et al., 2016; Moses et al., 2018), each of these feature vectors was constructed by concatenating high gamma z-score values for relevant electrodes across all of the time points in a time window relative to the target time point, capturing both spatial (multiple electrodes) and temporal (multiple time points) dynamics of the cortical activity (Fig. 4.8). For example, a feature vector associated with the speech event label at some time index t might consist of the neural data at time indices $\{t-10, t-9, \ldots, t+19, t+20\}$. These time windows were parameterized by the shift of the first time point relative to the time index t that it is associated with and by the duration of the time window (both were treated as hyperparameters). Prior to model fitting, all of the collected training data were restructured such that a spatiotemporal neural feature vector y_t and a speech event label h_t were available for every valid time point t.

To compute the speech event probabilities $p\left(h_t|q_t\right)$ at each time point, each speech event detection model applied linear discriminant analysis (LDA) to the leading principal components of the neural features. Using all of the available spatiotemporal neural feature vectors, we fit a principal component analysis (PCA) model with the constraint that the dimensionality of the projected feature vectors would be reduced to the minimum number of principal components required to explain a certain fraction of the variance across the features (this fraction was a hyperparameter determined during optimization). We then used these new projected feature vectors and the speech event labels to fit an LDA model implementing the least-squares solution with automatic shrinkage described by the Ledoit-Wolf lemma (Ledoit and Wolf, 2004). After training, these PCA-LDA models were capable of extracting the principal components from a previously unseen spatiotemporal feature vector and using the resulting projection to predict speech event probabilities (the LDA model assumed flat class priors when computing these probabilities). We used the Python package *scikit-learn* to implement the PCA and LDA models (Pedregosa et al., 2011).

During testing, the predicted speech event probabilities from these trained PCA-LDA models were used to detect the onsets and offsets of speech events (Fig. 4.9). For every time point t during neural data acquisition, the $p\left(h_t|y_t\right)$ probabilities were computed from the speech event probability model (Fig. 4.9a). For perception and production, these probabilities were smoothed using a sliding window average (Fig. 4.9b). Next, these smoothed probabilities were thresholded to either be 1 if the detection model suspected that the associated speech event type was occurring and 0 otherwise (Fig. 4.9c). These probability-thresholded binary values were then thresholded in time (debounced); a speech onset (or offset) was only detected if this binary value changed from 0 to 1 and remained 1 (or the opposite for offsets) for a pre-determined number of time points (Fig. 4.9d). Whenever a speech event offset was detected (which could only occur after an onset had been detected), the neural data segmented by the onset and offset were passed to the appropriate utterance classification model (question classification for perception events and answer classification for production events; Fig. 4.9e). The number of recent time points used during probability averaging, probability threshold value, time threshold duration, and onset and offset index shifts (integers added to the predicted onset and offset time indices before segmenting the neural data) were all treated as hyperparameters and set via optimization (with separate parameters for perception and production).

Utterance classification

For each subject and utterance type (questions and answers), we used utterance classification models to predict the likelihood of each utterance given a time segment of neural activity from the speech detector. For each utterance, we constructed an HMM to represent that utterance, with the phones comprising the utterance as hidden states and neural feature vectors as observed states. Given a time series of high gamma z-score values, each of these HMMs yielded the likelihood of observing those neural features during perception or production of the underlying phone sequence. These likelihoods are robust to natural variabilities in the durations of the phones in the sequence, which is a key motivation for using HMMs in this approach (even with a single speaker producing the same utterance multiple times, there will be small phone duration variabilities). We smoothed and normalized these phone sequence likelihoods to obtain the likelihood of each utterance given the neural features.

In each HMM, each hidden state q_t represented the phone that occurred at time index t within the corresponding utterance and each observation y_t was the neural feature vector associated with time index t. To construct each HMM, the representative phone sequence for the associated utterance was determined from the phonetic transcriptions. The transition matrix for that HMM, which specified the transition probabilities $p\left(q_{t+1}|q_t\right)$, was defined such that each hidden state was one of the phones in this sequence and could only self-transition (with some probability p_{self}) or transition to the next phone in the sequence (with probability $1 - p_{\text{self}}$). A self-transition probability of 1 was used for the final state. We used the silence phone token /sp/ as the initial and final states for each HMM. For example, the utterance "Cold" could be represented by a phone sequence of /sp k ˈoʊ l d sp/, and if the hidden state of the associated HMM was /l/ at some time index t, the hidden state could only be /l/ or /d/ at time index $t + 1$.

Similar to the relevant electrode channel selection used for the speech detection models (described in the previous section), we identified which channels should be considered relevant to the type of speech processing associated with each utterance type (perception processing for question utterances and production processing for answer utterances). Using the three previously described data subsets (one during production, one during perception, and one during silence), we performed two-tailed Welch's t-tests for each channel between the appropriate subsets for each utterance type (perception vs. silence for questions and production vs. silence for answers). Channels with a P-value less than a threshold hyperparameter value were considered relevant for the current utterance type and were used during subsequent phone likelihood modeling.

Also resembling the speech event modeling approach described in the previous section, PCA-LDA models were trained to compute the phone emission likelihoods $p\left(y_t|q_t\right)$ at each time point, with y_t denoting a spatiotemporal neural feature vector at time index t. The hyperparameters associated with these models, including the feature time window parameters and the PCA minimum variance fraction, were optimized separately from the parameters in the speech event model.

During testing, we used Viterbi decoding on each HMM to determine the likelihood of each utterance given a detected time segment of high gamma z-score values (Viterbi, 1967; Moses et al., 2016; Martin, 2017; Moses et al., 2018) (Fig. 4.10). Formally, we computed the log likelihood of each utterance using the following recursive formula:

$$v_{(t,s)} = w_e \log p\left(y_t|s\right) + \max_{i \in S}\left[v_{(t-1,i)} + \log p\left(s|i\right)\right], \tag{4.1}$$

where $v_{(t,s)}$ is the log probability of the most likely Viterbi path that ends in phone (state) s at time t, $p\left(y_t|s\right)$ is the phone emission likelihood (the probability of observing the neural feature vector y_t if the current phone is s), $p\left(s|i\right)$ is the phone transition probability (the probability of transitioning from phone i to phone s), w_e is an emission probability scaling factor (a model hyperparameter) to control the weight of the emission probabilities relative to the transition probabilities (see Section 4.6.1), and S is the set of all possible phones. To initialize the recursion, we forced each decoding to start with a Viterbi path log probability of zero for the first state (the initial silence phone /sp/) and negative infinity for every other state. After decoding for each HMM, the Viterbi path log probability at the final state and time point for that HMM represented the log likelihood ℓ_u of the corresponding utterance u given the neural data. Log probabilities are used here and in later computations for numerical stability and computational efficiency.

The computed log likelihoods for each utterance were then smoothed and normalized using the following formula:

$$\ell_u^* := \nu \ell_u - \log \left[\sum_{j \in U} \exp\left(\nu \ell_j\right) \right], \tag{4.2}$$

where ℓ_u^* is the smoothed and normalized log likelihood for utterance u, ν is the smoothing hyperparameter, and U is the set of all valid utterances (for the current utterance type). Because differences in utterance log likelihoods can be large (e.g., in the hundreds), the smoothing hyperparameter, which lay in the range $[0, 1]$, was included to allow the model to control how confident its likelihood predictions were. The closer ν was to zero, the smoother the log likelihoods were (less sample variance among the log likelihoods). The final log term in Eq. 4.2 represents the LogSumExp function and was used to compute the normalization constant for the current smoothed log likelihood values. After computing this constant and subtracting it from the smoothed log likelihoods, the ℓ_u^* values satisfied the following equality:

$$\sum_{j \in U} \exp\left(\ell_j^*\right) = 1. \tag{4.3}$$

These ℓ_u^* values were used as the utterance classification model's estimate of the utterance log likelihoods given the corresponding neural data.

Context integration

Because of the stimulus selection (each answer response was only valid for some of the questions) and task design (an answer response always followed a question presentation during testing) used in this work, we were able to design a context integration model that used predicted question likelihoods to update the predicted answer probabilities. Based on our previous demonstration of predicting auditory sentences from neural activity (Moses et al., 2018), we hypothesized that we could use reliable decoding of the questions in this task to improve the answer predictions, which was confirmed through our evaluations.

Prior to testing, we mathematically defined the relationship between the question and answer utterances in the form of conditional probabilities. These probabilities, referred to as the context priors, were computed using the following formula:

$$p\left(u_a | u_q\right) = \begin{cases} \frac{1}{N_{A,q}} & \text{if } u_a \text{ and } u_q \text{ are in same QA set} \\ 0 & \text{otherwise,} \end{cases} \tag{4.4}$$

where $p\left(u_a|u_q\right)$ is the context prior specifying the probability of responding to the question utterance u_q with the answer utterance u_a and $N_{A,q}$ is the number of answer utterances in the same QA set as u_q (the number of valid answer responses to u_q). The QA sets are given in Table 4.3. These context priors assume that the valid answer responses to any question are equally likely given the question.

During testing, the context integration model receives predicted utterance log likelihoods from both the question and answer classification models. Each time the model received predicted question log likelihoods (denoted $\ell^*_{U_Q}$, containing the log likelihoods $\ell^*_{u_q}$ for each question utterance u_q), it computed prior log probabilities for the answer utterances from these question likelihoods and the pre-defined context priors using the following formula:

$$\log p_{_Q}\left(u_a\right) = \log \left\{ \sum_{u_q \in U_Q} \exp\left[\log p\left(u_a \mid u_q\right) + \ell^*_{u_q}\right] \right\} + c, \tag{4.5}$$

where $p_{_Q}\left(u_a\right)$ is defined as the prior probability of the answer utterance u_a computed using $\ell^*_{U_Q}$, U_Q is the set of all question utterances, and c is some real-valued constant. Each time the model received predicted answer log likelihoods (the $\ell^*_{u_a}$ values for each answer utterance u_a), it computed posterior log probabilities for the answer utterances from these answer likelihoods and the answer priors. The unnormalized log posterior probabilities ϕ_{u_a} were computed for each answer utterance u_a using the following formula:

$$\phi_{u_a} := m \log p_{Q}(u_a) + \ell^*_{u_a} + d, \tag{4.6}$$

where m is the contextual prior scaling factor and d is some real-valued constant. Here, m is a hyperparameter that controls the weight of the answer priors relative to the answer likelihoods (a larger m causes the context to have a larger impact on the answer posteriors). We can then normalize these answer log posterior values using the following formula:

$$\phi^*_{u_a} := \phi_{u_a} - \log\left[\sum_{j \in U_A} \exp(\phi_j)\right], \tag{4.7}$$

where $\phi^*_{u_a}$ is the normalized log posterior probability of u_a and U_A is the set of all answer utterances. The constants c and d do not need to be computed in practice because they are canceled out during the normalization step in Eq. 4.7. These $\phi^*_{u_a}$ values satisfy the following equality:

$$\sum_{j \in U_A} \exp\left(\phi^*_j\right) = 1. \tag{4.8}$$

Finally, the utterance identities predicted by our system are computed as:

$$\hat{u}_q = \underset{u_q \in U_Q}{\operatorname{argmax}}\, \ell^*_{u_q}, \tag{4.9}$$

$$\hat{u}_{a-} = \underset{u_a \in U_A}{\operatorname{argmax}}\, \ell^*_{u_a}, \tag{4.10}$$

$$\hat{u}_{a+} = \underset{u_a \in U_A}{\operatorname{argmax}}\, \phi^*_{u_a}, \tag{4.11}$$

where $\hat{u}_q$, $\hat{u}_{a-}$, and $\hat{u}_{a+}$ are the system's predictions for questions, answers without context, and answers with context, respectively. The $\hat{u}_q$ and $\hat{u}_{a+}$ predictions are the system outputs during decoding, and the $\hat{u}_{a-}$ predictions are used in offline analyses. For a more thorough mathematical description of the context integration approach, see Section 4.6.1.

Although an answer response trial followed each question trial during testing, it was possible for the speech detector to fail to detect question or answer events (or to detect false positives). Because of this, we did not force the context integration model to always expect answer likelihoods after receiving question likelihoods or vice versa. Instead, during each test block, we maintained a set of values for the answer priors that were only updated when a new set of question likelihoods was received. When a new set of answer likelihoods was received, the current answer prior values were used to compute the posteriors. If answer likelihoods were received before receiving any question likelihoods, answer posteriors and answer with context predictions would not be computed from those likelihoods (although this did not occur in any of our test blocks).

4.5.7 Hyperparameter optimization

Each type of model (speech detection, utterance classification, and context integration) had one or more parameters that could not be learned directly from the training data. Instead of manually selecting values for these hyperparameters, we performed cross-validated hyperparameter optimization using the *hyperopt* Python package (Bergstra et al., 2011, 2013). This package uses a Bayesian-based optimization algorithm called the Tree-structured Parzen Estimator (sometimes referred to as the Tree of Parzen Estimators or abbreviated as TPE) to explore a hyperparameter space across multiple testing epochs. Although the full details of the algorithm are beyond the scope of this text, it operates by sampling hyperparameter values from pre-defined prior distributions, using a loss function to evaluate the current hyperparameters, and repeating these steps using knowledge gained from the evaluations it has performed. After a desired number of epochs, the hyperparameter set associated with the minimal loss value across all epochs is deemed the optimal hyperparameter set.

In this work, we performed hyperparameter optimization for each subject, model type, and test block. We used a leave-one-block-out cross-validation scheme for each test block. Specifically, during an optimization run for any given test block, the hyperparameters were evaluated on a held-out validation set comprised of all of the other test blocks available for the current subject. After optimization, we assessed the performance of our system using separate models for each test block with the corresponding optimal hyperparameters found during optimization. We used 250 epochs for each optimization run. All of the hyperparameters that were set via optimization are described in Table 4.4.

For each subject, we first performed optimization for the speech detection models. During each optimization epoch, the speech event probability model was trained using all of the available training data and tested on each block in the validation set. As described in Section 4.5.8, we used a custom speech detection score to evaluate the performance of the speech detector (a higher speech detection score signified better performance). The loss function used during speech detection optimization was defined as:

$$\mathcal{L}_{\text{detection}} := \sum_{\beta \in B, \psi \in \{\text{question}, \text{answer}\}} \left(1 - s^2_{\text{detection}, \beta, \psi}\right), \tag{4.12}$$

where $\mathcal{L}_{\text{detection}}$ is the detection loss, β signifies one of the blocks in the validation set B, ψ signifies one of the utterance types (either question or answer), and $s_{\text{detection}, \beta, \psi}$ is the speech detection score associated with validation block β and utterance type ψ. Thus, the optimal hyperparameters for the speech detection model associated with each test block were the hyperparameters that best detected the question and answer events in the validation blocks.

Next, we performed optimization for the utterance classification models. Separate optimizations were performed for the question and answer classifiers. During each optimization epoch, the phone likelihood models were trained using all of the available training data. Afterwards, the utterance classifiers predicted the utterance labels of the speech events that were detected by the optimized speech detection models in each validation block. We used cross entropy on the validation set as the loss function during optimization. Although the true speech event times were used during cross entropy calculations in other analyses, we chose to optimize the classifiers using the detected times to increase the robustness of the classifiers to imperfect speech event detection. To compute cross entropy on the decoded utterances from the detected events, we had to first convert the decoded sequence to classification trials. We performed this conversion by iterating through the actual utterances in chronological order and pairing each actual utterance label with the detected utterance label that had the closest detected speech offset time to the actual speech offset time (pairing a detected utterance label with more than one actual label was prevented). The optimal hyperparameters for each utterance classification model associated with each test block were the hyperparameters that resulted in the lowest cross entropy on the detected speech events in the validation blocks.

Finally, we optimized the context integration models. The only goal of this optimization process was to choose a value for the context prior scaling factor m. For each test block, we decoded utterance sequences in the validation blocks using the optimized speech detection and utterance classification models. During each optimization epoch, the answer with context predictions were computed using the decoded question and answer (without context) sequences and the current value of the hyperparameter m. Similar to the utterance classifier optimization, we used cross entropy on the validation set as the loss function during optimization. The decoded answer with context sequences were converted to classification trials so that the cross entropy could be computed. The optimal value of m for the context integration model associated with each test block was the value that resulted in the lowest cross entropy of the answer with context predictions using the detected speech events in the validation blocks.

Table 4.4: The description and optimization search space for each hyperparameter.

Optimization	Hyperparameter description	Search space type	Value range or choices
Speech Detection	Electrode relevance P-value threshold	Logarithmically uniform	$[10^{-50}, 10^{-3}]$
	Duration before t to include in the spatiotemporal neural feature vector y_t (in ms)	Uniform	$[1, 300]$
	Duration after t to include in the spatiotemporal neural feature vector y_t (in ms)	Uniform	$[1, 300]$
	Minimum amount of variance the principal components should explain when fitting the PCA model	Uniform	$[0.01, 0.99]$
	Question perception averaging window size (in samples)	Uniform (integer)	$[80, 160]$
	Question perception probability threshold	Uniform	$[0.4, 0.9]$
	Question perception time threshold (in samples)	Uniform (integer)	$[5, 60]$
	Question perception onset index shift (in samples)	Uniform (integer)	$[-100, 100]$
	Question perception offset index shift (in samples)	Uniform (integer)	$[-100, 300]$
	Answer production averaging window size (in samples)	Uniform (integer)	$[20, 80]$
	Answer production probability threshold	Uniform	$[0.4, 0.9]$
	Answer production time threshold (in samples)	Uniform (integer)	$[2, 10]$
	Answer production onset index shift (in samples)	Uniform (integer)	$[-100, 0]$
	Answer production offset index shift (in samples)	Uniform (integer)	$[-100, 50]$
Utterance classification (for questions and answers)	Electrode relevance P-value threshold	Logarithmically uniform	$[10^{-50}, 10^{-3}]$
	Set of hidden states for the HMMs (S)	Choice	Phones or phonemes[1]
	HMM self-transition probability (p_{self})	Uniform	$[0.1, 0.9]$
	Shift relative to t specifying the first data point in the spatiotemporal neural feature vector y_t (in ms)	Uniform	$[-200, 200]$
	Duration of each spatiotemporal neural feature vector (in ms)	Uniform	$[10, 400]$
	Minimum amount of variance the principal components should explain when fitting the PCA model	Uniform	$[0.01, 0.99]$
	Number of samples of each phone to include when training the phone likelihood models	Uniform (integer)	$[50, 3000]$
	HMM emission probability scaling factor (w_e)	Uniform	$[0.1, 5.0]$
	Log likelihood smoothing factor (ν)	Uniform	$[0.0001, 1.0]$
Context integration	Contextual prior scaling factor (m)	Logarithmically uniform	$[0.1, 10]$

[1] Phoneme labels were simply the phone labels without stress markings. Across all test blocks and subjects, phoneme labels were only deemed optimal for one question classifier and one answer classifier.

4.5.8 Evaluation methods

Primary evaluation metrics

We used the following metrics during the primary evaluations of our system: decoding accuracy rate, classification accuracy, cross entropy, speech detection score, and electrode discriminative power (Fig. 4.2). The decoding accuracy rate metric represented the full performance of the system (the combined performance of the speech detection, utterance classification, and context integration models). When computing the accuracy rates, we first computed the utterance error rate for each prediction type and each test block. The utterance error rate, an analog of the commonly-used word error rate metric, is a measure of the edit (Levenshtein) distance between the actual and decoded utterance label sequences in a given test block. The accuracy rate was then computed as $\max\left[0, 1 - (\text{utterance error rate})\right]$. For each subject and prediction type, the mean and variance of the accuracy rate was calculated using the accuracy rates for each test block.

To compute the classification accuracy and cross entropy metrics on a set of results for any subject, we obtained utterance classification results by re-evaluating our system using only the utterance classification and context integration models (and not the speech detection models). In this approach, we performed decoding on the test blocks using the actual speech event times and the previously trained utterance classification models. This was equivalent to performing decoding with a speech detection model that perfectly detected each speech event. We decremented each speech onset time and incremented each speech offset time by 300 ms to include silence time points before and after the utterance in each speech-related time window of neural data passed to the classifiers. We then performed context integration model optimization with these new classification results and applied the optimized context integration models to these results. After this step, we then pooled all of the pairs of actual and predicted utterance labels for each prediction type (questions, answers without context, and answers with context) across all of the test blocks for each subject.

The classification accuracy measured the fraction of these classification trials in which the utterance classification model correctly predicted the identity of the utterance. To compute the classification accuracy for a given subject and prediction type, we created an array of indicator variables that contained 1 for each classification trial in which the predicted and actual labels were equal and 0 for the remaining trials. We then calculated the mean and variance of the classification accuracy as the mean and variance of this indicator variable array.

The cross entropy metric quantified the amount of predictive information provided by the utterance classification and context integration models during testing and hyperparameter optimization. We computed the cross entropy values using the surprisal values for each classification trial, prediction type, and subject. For a given trial and prediction type, the relevant surprisal value for that trial is equal to the negative of the predicted log probability associated with the actual utterance label. The cross entropy is equal to the mean of these surprisals, and we also used the variance of these surprisals as a proxy for the variance of the cross entropy. Lower cross entropy indicates better performance.

We used a custom speech detection score metric to measure the performance of the speech detection model during testing and hyperparameter optimization. This metric was computed as a weighted combination of a frame-by-frame accuracy value a_{frame} and a general event detection accuracy value a_{event}. The frame-by-frame accuracy measured the performance of the speech detector using the detected presence or absence of a speech event at each time point. This measure resembles sensitivity and specificity analyses used commonly for screening test evaluations involving true/false positives/negatives. For any test block and utterance type, true positives were time points (frames) in that test block during which the speech detector correctly predicted that a speech event was occurring, true negatives were time points during which the speech detector correctly predicted that silence was occurring, and false positives and negatives were time points in which the speech detector made an incorrect prediction. The phonetic transcriptions were used to determine the actual times of the speech events. When using these transcribed speech times, we decremented each speech onset time and incremented each speech offset time by 300 ms to label some silence time points before and after each utterance as positive frames. We performed this modification to encourage the optimizer to select hyperparameters that would include silence before and after each utterance in the detected neural feature time windows, which is useful during utterance classification. We calculated the frame-by-frame accuracy measure using the following formula:

$$a_{\text{frame}} = \frac{w_{\text{P}} N_{\text{TP}} + (1 - w_{\text{P}}) N_{\text{TN}}}{w_{\text{P}} N_{\text{P}} + (1 - w_{\text{P}}) N_{\text{N}}}, \tag{4.13}$$

where w_{P} is the positive weight fraction, N_{TP} is the number of true positives detected, N_{TN} is the number of true negatives detected, N_{P} is the total number of positive frames in the test data, and N_{N} is the total number of negative frames in the test data. The positive weight fraction was included to allow control over how important true positive detection was relevant to true negative detection. In practice, we used $w_{\text{P}} = 0.75$, meaning that correctly detecting positive frames was three times as important as correctly detecting negative frames. We used this value to encourage the optimizer to select hyperparameters that would prefer to make false positive errors than false negative errors, since the performance of the utterance classifiers should diminish more heavily if a few speech-relevant time points were excluded from the detected time window than if a few extra silence time points were included. The general event detection accuracy measured how well the speech events were detected without considering which time points were associated with each event. For any test block and utterance type, the general event detection accuracy value was computed using the following formula:

$$a_{\text{event}} = 1 - \min\left(1, \frac{|N_{\text{DE}} - N_{\text{AE}}|}{N_{\text{AE}}}\right), \tag{4.14}$$

where N_{DE} and N_{AE} are the number of detected and actual speech events in the current test block, respectively. To compute the speech detection score $s_{\mathrm{detection}}$, these two measures were combined using the following formula:

$$s_{\mathrm{detection}} = w_{\mathrm{F}} a_{\mathrm{frame}} + (1 - w_{\mathrm{F}})\, a_{\mathrm{event}}, \tag{4.15}$$

where w_{F} is the frame-by-frame accuracy weight fraction, which allows control over how much impact the frame-by-frame accuracy measure has on the speech detection score relative to the general event detection accuracy. In practice, we let $w_{\mathrm{F}} = 0.5$ for an equal weighting between the two measures.

To assess the importance of each electrode during phone and speech event likelihood modeling, we estimated the discriminative power of each electrode within the trained PCA-LDA models (Moses et al., 2016). To compute these discriminative powers, we first arbitrarily selected a test block for each subject and obtained the trained and optimized utterance classification and speech detection models associated with that test block. For each of these models, we examined the learned parameters within the internal LDA model. For each feature in the LDA model (which is a principal component value from the PCA model), we measured the between-class variance for that feature by computing the variance of the corresponding class means. We used the values along the diagonal of the shared covariance matrix as a measure of the within-class variance of each feature (because we did not force diagonal covariance matrices in the LDA models, this is only an approximation of the true within-class variances). We then estimated the discriminative power for each LDA feature using the following formula (which resembles a coefficient of determination calculation):

$$\eta_i = 1 - \frac{\sigma^2_{\mathrm{w},i}}{\sigma^2_{\mathrm{w},i} + \sigma^2_{\mathrm{b},i}}, \tag{4.16}$$

where η_i, $\sigma^2_{\mathrm{w},i}$, and $\sigma^2_{\mathrm{b},i}$ are the estimated discriminative power, within-class variance, and between-class variance, respectively, for the ith LDA feature. To obtain the discriminative powers for each original feature in the spatiotemporal neural feature vectors (the inputs to the PCA model), the absolute values of the PCA component weights were used to project the LDA feature discriminative powers back into the original feature space. Finally, the discriminative power for each electrode was set equal to the maximum discriminative power value observed among the original features associated with that electrode (that is, the maximum function was used to aggregate the discriminative powers across time for each electrode within the spatiotemporal feature vectors). The resulting discriminative power values can be used to quantify the relative contributions made by each electrode during phone or speech event discrimination.

Utterance classifier sensitivity analyses

We investigated the sensitivity of the utterance classifiers to limited data availability and sub-optimal hyperparameter configurations (Fig. 4.3). To assess how the amount of available training data affected classification performance, we evaluated classifiers that were trained on varying amounts of data. For each subject and utterance type, we first randomly selected $n = 1$ samples of each utterance from the available training data. For each utterance selected this way, we obtained the neural feature vectors and phone labels that occurred between 150 ms before the speech onset and 150 ms after the speech offset. The onset and offset times were determined from the phonetic transcriptions, and the time points before and after each utterance were included so that the classifiers would have sufficient samples of the silence phone /sp/. Utterance classification models were trained using these features and labels associated with the selected utterances (separate models were trained for each test block using the optimized hyperparameter values for that block). We then evaluated the classifiers (and the context integration models) across all of the test blocks using the classification accuracy and cross entropy metrics. We repeated this process 15 times, each time drawing a new random selection of n samples of each utterance to use during training. Afterwards, we then performed all of these steps for every integer value of n in the range $[2, N_{\text{max}}]$, where N_{max} denotes the total number of samples of each utterance available across the training blocks for the subject ($N_{\text{max}} = 10, 30, 20$ for subjects 1–3). All random selections of the utterances to use during training were sampled without replacement. We plotted the mean and SEM of the classification accuracy and cross entropy values across the 15 repeats for each value of n to visualize how classification was affected by the amount of training data.

To better understand the effect that hyperparameter selection had on classifier performance, we evaluated classifiers with many different hyperparameter configurations. For each subject, we arbitrarily selected one of the test blocks for that subject and obtained each of the 250 hyperparameter configurations that were evaluated (on a separate validation set) during optimization of the question and answer classifiers for that test block. For each of these hyperparameter configurations, we trained the utterance classifiers with the configuration using all of the training data available for that subject, and we then evaluated the classifier performance on the chosen test block using the classification accuracy and cross entropy metrics. We plotted the resulting accuracies and cross entropies for each of these hyperparameter configurations (which also included the configuration that was deemed optimal).

We also characterized the performance of the utterance classification and context integration models while limiting the cortical regions that each model had access to during training and testing. For each subject, we separated all of the available electrodes for that subject into two subsets: electrodes below the Sylvian fissure (infra-Sylvian) and electrodes above the Sylvian fissure (supra-Sylvian). We then fit question classification models using only the neural data recorded from infra-Sylvian electrodes and answer classification models using only the neural data recorded from supra-Sylvian electrodes. For each of these models, the relevant channel selection, hyperparameter optimization, electrode discriminative power calculation, and training and testing procedures were similar to the approaches used for the standard utterance classification models (with the only meaningful difference being the electrode subsets used for each model). The context integration model used the question likelihoods from the infra-Sylvian model and the answer likelihoods (without context) from the supra-Sylvian model to compute the answer posteriors. We plotted the performance of these models across all of the test blocks for each subject using the classification accuracy and cross entropy metrics.

Utterance classifier characterization

To better understand how the utterance classifiers made their predictions, we performed additional analyses on the Viterbi decoding and phone likelihood modeling approaches used by the answer classifiers (Fig. 4.4). To visualize the process by which the classifiers used Viterbi decoding to update the predicted likelihood of each utterance as it received additional neural data, we examined how the utterance likelihoods changed over time during a correctly predicted answer utterance for one of the subjects. Using the time window of neural activity associated with that utterance and the trained phone likelihood model for the test block containing the selected trial, we performed Viterbi decoding on the HMM for each answer utterance. At each time index t during the Viterbi decoding with each HMM, we stored the log likelihood of the most likely Viterbi path through the HMM at that time index given the neural feature vectors $\{y_0, y_1, \ldots, y_t\}$ (where y_0 is the first feature vector in the time window). Afterwards, the path likelihoods at each time point were smoothed (using ν, the smoothing hyperparameter from the classifier) and normalized (to sum to 1) across all utterances. The resulting values were plotted as the probability of each utterance at each time point during Viterbi decoding.

To assess how well the answer phone likelihood models were able to discriminate between the phonetic classes, we computed phone confusions across all of the test blocks and subjects. For each test block, we used the phone likelihood model within the associated answer classification model to predict the phone label at each time point. We compared the predicted and actual phone labels at each time point across all test blocks to compute the plotted phone confusion matrix. We excluded one test block for subject 2 from this analysis because the answer classification model associated with that test block used phonemic labels (labels without stress markers) instead of the phonetic labels used in all of the other test blocks (see Table 4.4).

We also used these path probabilities to measure the amount of time points required before each classifier finalized its prediction of which utterance was most likely during each trial. Across all test blocks and subjects, we computed the Viterbi path probabilities at each time point during classification of each answer trial. For each trial, we used these path probabilities to find the earliest time index at which the predicted utterance (the utterance with the highest path probability at the final time index) became and remained more likely than all of the other utterances (denoted $t_{\text{finalization}}$). We computed the decision finalization time for a trial using the following formula:

$$\tau = \frac{t_{\text{finalization}} - t_{\text{onset}}}{t_{\text{offset}} - t_{\text{onset}}}, \tag{4.17}$$

where τ is the decision finalization time and t_{onset} and t_{offset} are the speech onset and offset time indices of the utterance (obtained from the phonetic transcriptions). To assess how well these decision finalization times could be explained by the phonetic content and pronunciation of the stimuli, we also computed these decision finalization times using phone likelihoods constructed directly from the phonetic transcriptions (without using neural data to infer the phone likelihoods). For each trial, we provided as input to the Viterbi decoder for each HMM a time series of phone likelihoods in which, at each time point, the probability of the phone that was actually occurring (according to the phonetic transcriptions) was equal to 0.9 and the remaining 0.1 probability mass was divided evenly among the other phones. During Viterbi decoding, all of the non-zero phone transition probabilities $p\left(q_{t+1}|q_t\right)$ for each

HMM were set equal to 0.5. To compare the finalization times between the neural-based and transcription-based analyses, we restricted the trials that were considered to only contain trials in which the neural-based classifier correctly predicted the utterance identity (we did not find a significant difference in finalization times between the correct and incorrect trials, $P = 0.37$, two-tailed Welch's t-test). In the decision finalization time plot, trials in which the finalization time was negative for the neural-based model were excluded (5 trials were excluded from and 89 trials were included in the figure).

4.5.9 Statistical testing

The statistical tests used during evaluation of our results are described in this section. For all tests, we considered P-values less than 0.05 as significant.

When comparing the decoding accuracy rates to chance (Fig. 4.2a, Fig. 4.5a), we used one-tailed bootstrap tests on individual test blocks and then combined the test results across these blocks. First, for each subject, test block, and prediction type (questions, answers without context, and answers with context), we determined the number of actual utterance labels in the current block. We then created a sequence with length equal to this number and with elements randomly sampled from the set of possible utterance labels for the current prediction type (sampling was done with replacement and with a uniform probability distribution across the possible labels). Next, we computed the decoding accuracy rate by comparing this random sequence with the actual sequence. We performed this process of creating a random sequence and computing its accuracy rate one million times. Afterwards, we created a normal distribution parameterized by the mean and standard deviation of the accuracy rates observed during this process. We determined the value of the cumulative distribution function (CDF) of this distribution at the value equal to the decoding accuracy rate associated with the current test block and prediction type. The one-tailed P-value for the null hypothesis that the decoded accuracy rate was not above chance was equal to 1 minus this CDF value. After computing these P-values for each test block for a subject, the final P-value for the subject was computed by combining the individual P-values across blocks using Fisher's method (sometimes referred to as Fisher's combined probability test) (Fisher, 1932). Our method of measuring chance performance was arguably an overestimate of the true chance performance because it uses the exact same sequence length as the actual utterance sequence within a test block (this is equivalent to assuming that the speech detector always detected the correct number of events).

When comparing the answer with context and answer without context decoding accuracy rates (Fig. 4.2a, Fig. 4.5a), we used one-tailed permutation tests on individual test blocks and then combined the test results across these blocks. First, for each subject and test block, we obtained the decoded with context and without context answer sequences. These sequences were always the same length (each of these types of predictions were made every time an answer event was detected during testing). We then created a mixture predicted sequence of the same length as these sequences in which the value at any index i in this sequence was randomly selected as either the utterance label at index i in the without context decoded sequence or the label at index i in the with context decoded sequence (with an equal probability of choosing from either). Next, we computed the decoding accuracy rate for this mixture sequence. We performed this process of randomly creating a mixture sequence and computing its accuracy rate one million times. Afterwards, we created a normal distribution parameterized by the mean and standard deviation of the accuracy rates observed during this process. We determined the value of the CDF of this distribution at the value equal to the decoding accuracy rate associated with the answer with context predictions in the current test block. The one-tailed P-value for the null hypothesis that the accuracy rate of the answer predictions with context was not above the predictions without context was equal to 1 minus this CDF value. After computing these P-values for each test block for a subject, we used Fisher's method to combine these individual P-values across blocks and obtain the final P-value for the subject.

When comparing the classification accuracies to chance (Fig. 4.2b, Fig. 4.3d, Fig. 4.5b, Fig. 4.6d), we used a one-tailed bootstrapped Welch's t-test. First, for each subject and prediction type, we computed an indicator variable array that contained 1 for each classification trial (across all test blocks) in which the predicted and actual labels were equal and 0 for the remaining trials. Then, we created an array with length equal to the actual number of trials across all test blocks. Each element in this array was randomly sampled from the set of possible utterance labels for the current prediction type (sampling was done with replacement and with a uniform probability distribution across the possible labels). We the computed the chance classification accuracy by comparing these random labels to the actual labels, generating an indicator variable array similar to the ones created for the predictions. We performed this process of creating a random label array and computing its classification accuracy one million times. Afterwards, we performed a one-tailed Welch's t-test between the predicted and chance indicator arrays (with the sample size equal to the number of trials for the predictions and equal to one million times the number of trials for chance). The resulting P-value was the probability of the null hypothesis that the predicted classification accuracies were not higher than chance.

When comparing the answer with context and answer without context classification accuracies (Fig. 4.2b, Fig. 4.3d, Fig. 4.5b, Fig. 4.6d), we used a one-tailed exact McNemar's test. First, for each subject, we obtained the indicator variable arrays described earlier for the with context and without context answer predictions. We then used a one-tailed exact McNemar's test to compare these two arrays (McNemar, 1947). McNemar's test is suited for comparing two paired binary sequences. In this application of McNemar's test, the number of trials that were correctly predicted when using context but incorrect without context are compared to the number of trials that were correctly predicted without using context but incorrect with context. The resulting P-value from this test was the probability of the null hypothesis that the classification accuracy was not higher for answer predictions with context than those without context.

When comparing cross entropies to chance (Fig. 4.2c, Fig. 4.3d, Fig. 4.5c, Fig. 4.6d), we used a one-tailed bootstrap test. First, for each subject and prediction type, we computed the negative predicted log probability values associated with the actual utterance label within each classification trial (across all test blocks). These negative log probability values are referred to as surprisals. Then, we created a new array with length equal to the number of trials across all test blocks. Each element in this array was randomly sampled from the array of surprisal values associated with the predictions (sampling was done with replacement and with a uniform probability distribution across all of the surprisals). We then computed the chance cross entropy by taking the mean of these randomly-sampled surprisals. We performed this process of randomly sampling the surprisals and computing the cross entropy one million times. Afterwards, we created a normal distribution parameterized by the mean and standard deviation of the chance cross entropy values. We determined the value of the CDF of this distribution at the value equal to the chance cross entropy value, which was computed as the negative log of 1 divided by the number of possible labels for the current prediction type. The one-tailed P-value for the null hypothesis that the predicted cross entropy was not lower than chance was equal to 1 minus this CDF value (lower cross entropy indicates better performance).

When comparing the answer with context and answer without context cross entropies (Fig. 4.2c, Fig. 4.3d, Fig. 4.5c, Fig. 4.6d), we used a one-tailed Wilcoxon signed-rank test. First, for each subject, we obtained the surprisal arrays described earlier for the with context and without context answer predictions. We then used a one-tailed Wilcoxon signed-rank test to compare these paired samples. The resulting P-value from this test was the probability of the null hypothesis that the cross entropy was not lower for answer predictions with context than those without context.

When comparing the decision finalization times for the answer classifiers to the speech offset time (Fig. 4.4c), we used a one-tailed single-sample Wilcoxon signed-rank test. This test was performed on an array created by subtracting each decision finalization time (scaled such that 0 was the speech onset time and 1 was the speech offset time) from the speech offset time (which was 1 due to this scaling). The resulting P-value from this test was the probability of the null hypothesis that the decision finalization times for the answer classifiers did not occur before the speech offset.

When comparing the neural-based and transcription-based decision finalization times (Fig. 4.4c), we used a two-tailed Wilcoxon signed-rank test. This test was performed using the paired decision finalization time samples (a neural-based and transcription-based finalization time was available for each trial used in this test). The resulting P-value from this test was the probability of the null hypothesis that the neural-based and transcription-based finalization times were both sampled from the same underlying distribution.

4.5.10 Real-time processing setup

In our previous work, we introduced the *rtNSR* software package and used it to classify perceived sentences using human cortical activity in real-time (Moses et al., 2018). Written in Python (Python Software Foundation, 2016), this package is flexible and efficient due to its modular structure and utilization of software pipelining (Lam, 1988). After further development of the package, we used it in this work to present the audio and visual stimuli, process the neural signals, and perform speech decoding in real-time. We also used it for offline model training and data analysis.

An overview of the *rtNSR* processing flow used during real-time testing is given in Fig. 4.11. Signals from the DAQ rig were streamed into the Linux machine running *rtNSR* using a real-time interface card (PO8e, Tucker-Davis Technologies). These signals were then passed through the previously described digital filter chain to extract a measure of the high gamma analytic amplitude from each channel at 95.37 Hz. After another process normalizes the analytic amplitudes, a separate process obtains the high gamma z-score values and performs speech event detection on the signals in a sliding window fashion. Whenever an event is detected, the neural data associated with that event is passed to the appropriate (either question or answer) classifier process, which uses phone-level Viterbi decoding to output likelihoods over the corresponding utterances. An utterance predictor process obtains these likelihoods and uses them to update the answer priors, perform context integration, and output the decoded utterances to a separate process that displays the results. Throughout all of these steps, a GUI process handles the presentation of visual and acoustic stimuli to the subject. The behavioral metadata (e.g., stimulus times) are stored along with the high gamma signals (prior to normalization) to disk on the real-time computer.

Due to clinical time constraints, we were not able to perform hyperparameter optimization prior to real-time testing with the subjects. All of the results reported in this work were computed using offline simulations of the data with the *rtNSR* system, a process that we described in our previous work. During the offline simulations, the real-time process that reads samples from the real-time interface card is replaced with a process that simulates input samples from a dataset on disk. The remainder of the decoding pipeline remains the same. During online testing at the patient's bedside, the system performed decoding without experiencing systematic/runtime errors and with negligible latency using hyperparameter values chosen via trial and error on datasets that were previously collected. Therefore, we can reasonably expect that the decoding results we observe in our offline simulations would have been identical to those in the online setting with the patients, since the only differences between the online and offline tests were the specific values of the hyperparameters.

4.6 Supplementary data

4.6.1 Supplementary notes

Supplementary note 1. Likelihood normalization and the emission probability scaling factor w_e.

In theory, the HMMs used in the utterance classification models require the likelihood values $p\left(y_t|q_t\right)$. In practice, however, we obtained phone posteriors $p\left(q_t|y_t\right)$ from the LDA models and used these in place of the likelihoods. Because we used flat (uniform) priors over the phone classes in these models, these posteriors were simply equal to the likelihoods after being scaled by an unknown normalization constant. This can be shown via Bayes' rule:

$$p\left(q_t|y_t\right) = \frac{p\left(y_t|q_t\right) p\left(q_t\right)}{p\left(y_t\right)} = Zp\left(y_t|q_t\right), \tag{4.18}$$

where $p\left(q_t\right)$ is a constant because flat priors were used and Z is the unknown constant caused by the presence of the $p\left(y_t\right)$ term and the $p\left(q_t\right)$ constant.

This discrepancy is addressed by the emission probability scaling factor w_e. By including this hyperparameter, the contribution of the emission probabilities during each iteration of Viterbi decoding (in Eq. 4.1 in the main text) becomes $w_e Z p\left(y_t|q_t\right)$. Because the value of w_e is set through hyperparameter optimization, the impact that this constant Z has on decoding is mitigated. This assumes that the optimizer is capable of finding a satisfactory value of this hyperparameter within its pre-defined range of possible values, which we have observed in practice.

Supplementary note 2. Mathematical formulation of the context integration model.

During testing, the utterance classification models receive (from the speech detection model) the high gamma features associated with detected question (γ_Q) and detected answer (γ_A) events. A primary goal of these classifiers and the context integration model is to predict the most likely answer utterance $\hat{u}_{a+}$ given the neural features γ_Q and γ_A. This goal can be expressed as:

$$\hat{u}_{a+} = \underset{u_a \in U_A}{\operatorname{argmax}}\, p\left(u_a \,|\, \gamma_Q, \gamma_A\right), \tag{4.19}$$

where u_a is one of the answer utterances and U_A is the set of all answer utterances.

This conditional probability $p\left(u_a \,|\, \gamma_Q, \gamma_A\right)$ represents the posterior probability of u_a given the question-related and answer-related neural features. We can refactor this posterior probability using the following steps (a description of each step is provided after the equations):

$$\begin{aligned}
p\left(u_a \mid \gamma_Q, \gamma_A\right) &= \sum_{u_q \in U_Q} p\left(u_a, u_q \mid \gamma_A, \gamma_Q\right) && (4.20)\\
&= \sum_{u_q \in U_Q} p\left(u_a \mid u_q, \gamma_A, \gamma_Q\right) p\left(u_q \mid \gamma_A, \gamma_Q\right) && (4.21)\\
&= \sum_{u_q \in U_Q} p\left(u_a \mid u_q, \gamma_A\right) p\left(u_q \mid \gamma_A, \gamma_Q\right) && (4.22)\\
&= \sum_{u_q \in U_Q} \frac{p\left(\gamma_A \mid u_a, u_q\right) p\left(u_a \mid u_q\right)}{p\left(\gamma_A \mid u_q\right)} p\left(u_q \mid \gamma_Q, \gamma_A\right) && (4.23)\\
&= \sum_{u_q \in U_Q} \frac{p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right)}{p\left(\gamma_A \mid u_q\right)} p\left(u_q \mid \gamma_Q, \gamma_A\right) && (4.24)\\
&= \sum_{u_q \in U_Q} \frac{p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right)}{p\left(\gamma_A \mid u_q\right)} \frac{p\left(\gamma_A, \gamma_Q \mid u_q\right) p\left(u_q\right)}{p\left(\gamma_A, \gamma_Q\right)} && (4.25)\\
&= \sum_{u_q \in U_Q} \frac{p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right)}{p\left(\gamma_A \mid u_q\right)} \frac{p\left(\gamma_A \mid u_q\right) p\left(\gamma_Q \mid u_q\right) p\left(u_q\right)}{p\left(\gamma_A, \gamma_Q\right)} && (4.26)\\
&= \sum_{u_q \in U_Q} p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right) \frac{p\left(\gamma_Q \mid u_q\right) p\left(u_q\right)}{p\left(\gamma_A, \gamma_Q\right)} && (4.27)\\
&= \frac{1}{p\left(\gamma_A, \gamma_Q\right)} \sum_{u_q \in U_Q} p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right) p\left(\gamma_Q \mid u_q\right) p\left(u_q\right) && (4.28)\\
&= \frac{1}{p\left(\gamma_A, \gamma_Q\right) \left|U_Q\right|} \sum_{u_q \in U_Q} p\left(\gamma_A \mid u_a\right) p\left(u_a \mid u_q\right) p\left(\gamma_Q \mid u_q\right) && (4.29)\\
&= \frac{p\left(\gamma_A \mid u_a\right)}{p\left(\gamma_A, \gamma_Q\right) \left|U_Q\right|} \sum_{u_q \in U_Q} p\left(u_a \mid u_q\right) p\left(\gamma_Q \mid u_q\right) && (4.30)\\
&\propto p\left(\gamma_A \mid u_a\right) \sum_{u_q \in U_Q} p\left(u_a \mid u_q\right) p\left(\gamma_Q \mid u_q\right) && (4.31)
\end{aligned}$$

Each step in the above formulation is described below:

- 4.20: The posterior probability can be expressed as the sum of the joint probability of the answer utterance u_a and question utterance u_q for each question utterance in the set of all question utterances U_Q (while still conditioned on the neural responses).
- 4.21: The probability can be refactored using the chain rule of probability.

- 4.22: u_a is independent of γ_Q given u_q.
- 4.23: The first probability term is refactored using Bayes' theorem.
- 4.24: γ_A is independent of u_q given u_a.
- 4.25: The second probability term is refactored using Bayes' theorem.
- 4.26: One of the terms is refactored into two terms using the fact that γ_A and γ_Q are conditionally independent given u_q.
- 4.27: A term in the numerator of one fraction cancels the identical term in the denominator of the other fraction.
- 4.28: The term in the denominator of the remaining fraction can be moved outside of the sum because it does not depend on u_q.
- 4.29: Because we assume a uniform prior over the question utterances, the $p\left(u_q\right)$ term is a constant value equal to 1 divided by the total number of question utterances and can be moved outside of the sum because it does not depend on u_q.
- 4.30: The first term in the sum is moved outside of the sum since it does not depend on u_q.
- 4.31: The denominator of the fraction outside of the sum does not depend on u_a, so the posterior probability can be expressed as being proportional to the remaining terms.

The terms in Eq. 4.31 are defined below:

- $p\left(\gamma_A \,|\, u_a\right)$ represents the answer likelihoods obtained from the answer classifier.
- $p\left(\gamma_Q \,|\, u_q\right)$ represents the question likelihoods obtained from the question classifier.
- $p\left(u_a \,|\, u_q\right)$ represents the pre-defined context priors.
- $\sum_{u_q \in U_Q} p\left(u_a \,|\, u_q\right) p\left(\gamma_Q \,|\, u_q\right)$ represents the answer priors.

In practice, we performed the calculations using log probabilities, and we used a contextual prior scaling factor m to control the weight of the answer priors relative to the answer likelihoods when computing the answer posteriors. With these modifications, the following formulas can be used to define the unnormalized answer log posterior probabilities:

$$p\left(u_a \mid \gamma_Q, \gamma_A\right) \propto p\left(\gamma_A \mid u_a\right) \left[\sum_{u_q \in U_Q} p\left(u_a \mid u_q\right) p\left(\gamma_Q \mid u_q\right) \right]^m , \tag{4.32}$$

$$\begin{aligned}
\phi_{u_a} &:= \log p\left(u_a \mid \gamma_A, \gamma_Q\right) && (4.33)\\
&= \log p\left(\gamma_A \mid u_a\right) + m \log \left\{ \sum_{u_q \in U_Q} \exp\left[\log p\left(u_a \mid u_q\right) + \log p\left(\gamma_Q \mid u_q\right)\right] \right\} + \kappa && (4.34)\\
&= \ell^*_{u_a} + m \log \left\{ \sum_{u_q \in U_Q} \exp\left[\log p\left(u_a \mid u_q\right) + \ell^*_{u_q}\right] \right\} + \kappa, && (4.35)
\end{aligned}$$

Each of these additional formula are described below:

- 4.32: In practice, the formula representing the answer posteriors includes the contextual prior scaling factor m.
- 4.33: ϕ_{u_a} is defined as the unnormalized log posterior probability of answer utterance u_a given the neural data γ_Q and γ_A.
- 4.34: When re-factoring a proportionality equation to an equality using log, a constant scalar value κ is introduced.
- 4.35: Using notation introduced in the main text (in Eq. 4.2), we use $\ell^*_{u_a}$ to denote the log likelihood of utterance u_a obtained from the answer classifier and $\ell^*_{u_q}$ to denote the log likelihood of utterance u_q obtained from the question classifier.

In practice, we do not compute the value of κ. We can define a variable to represent the unnormalized log posterior values without κ:

$$\phi'_{u_a} = \phi_{u_a} - \kappa = \ell^*_{u_a} + m \log \left\{ \sum_{u_q \in U_Q} \exp \left[\log p\left(u_a \mid u_q\right) + \ell^*_{u_q} \right] \right\} \tag{4.36}$$

We can then predict the most likely answer utterance $\hat{u}_{a+}$ directly from these ϕ'_{u_a} values:

$$\hat{u}_{a+} = \underset{u_a \in U_A}{\operatorname{argmax}} \, \phi'_{u_a}. \tag{4.37}$$

Here, κ does not need to be included because it will not affect which answer utterance was most likely.

Although we did not need to normalize the answer log posteriors to predict the most likely answer utterance, we still require normalized log posteriors (normalized to sum to 1 across all answer utterances) when calculating the cross entropy of the answer with context predictions. We compute normalized answer log posteriors using the following formulation:

$$\begin{aligned}
\phi^*_{u_a} &:= \phi_{u_a} - \log \left[\sum_{j \in U_A} \exp\left(\phi_j\right) \right] && (4.38) \\
&= \phi'_{u_a} + \kappa - \log \left[\sum_{j \in U_A} \exp\left(\phi'_j + \kappa\right) \right] && (4.39) \\
&= \phi'_{u_a} + \kappa - \log \left[\exp\left(\kappa\right) \sum_{j \in U_A} \exp\left(\phi'_j\right) \right] && (4.40) \\
&= \phi'_{u_a} + \kappa - \log \left[\exp\left(\kappa\right) \right] - \log \left[\sum_{j \in U_A} \exp\left(\phi'_j\right) \right] && (4.41) \\
&= \phi'_{u_a} - \log \left[\sum_{j \in U_A} \exp\left(\phi'_j\right) \right] && (4.42)
\end{aligned}$$

Each of these steps is described below:

- 4.38: $\phi^*_{u_a}$ is defined as the normalized log posterior probability of answer utterance u_a given the neural data γ_Q and γ_A (the term on the right represents the LogSumExp function used to normalize log probabilities).

- 4.39: We can replace the ϕ_{u_a} terms using our definition of ϕ'_{u_a}.
- 4.40: Because of the associativity of multiplication, we can express $\exp\left(\phi'_j + \kappa\right)$ as the product of $\exp(\kappa)$ and $\exp(\phi_j)$ and then move $\exp(\kappa)$ out of the sum because it does not depend on j.
- 4.41: From the logarithmic identity for the logarithm of a product, we can separate the terms in the log function into the sum of the logarithms of the individual terms.
- 4.42: The $-\log[\exp(\kappa)]$ term simplifies to $-\kappa$, which cancels out the κ term.

4.6.2 Supplementary figures

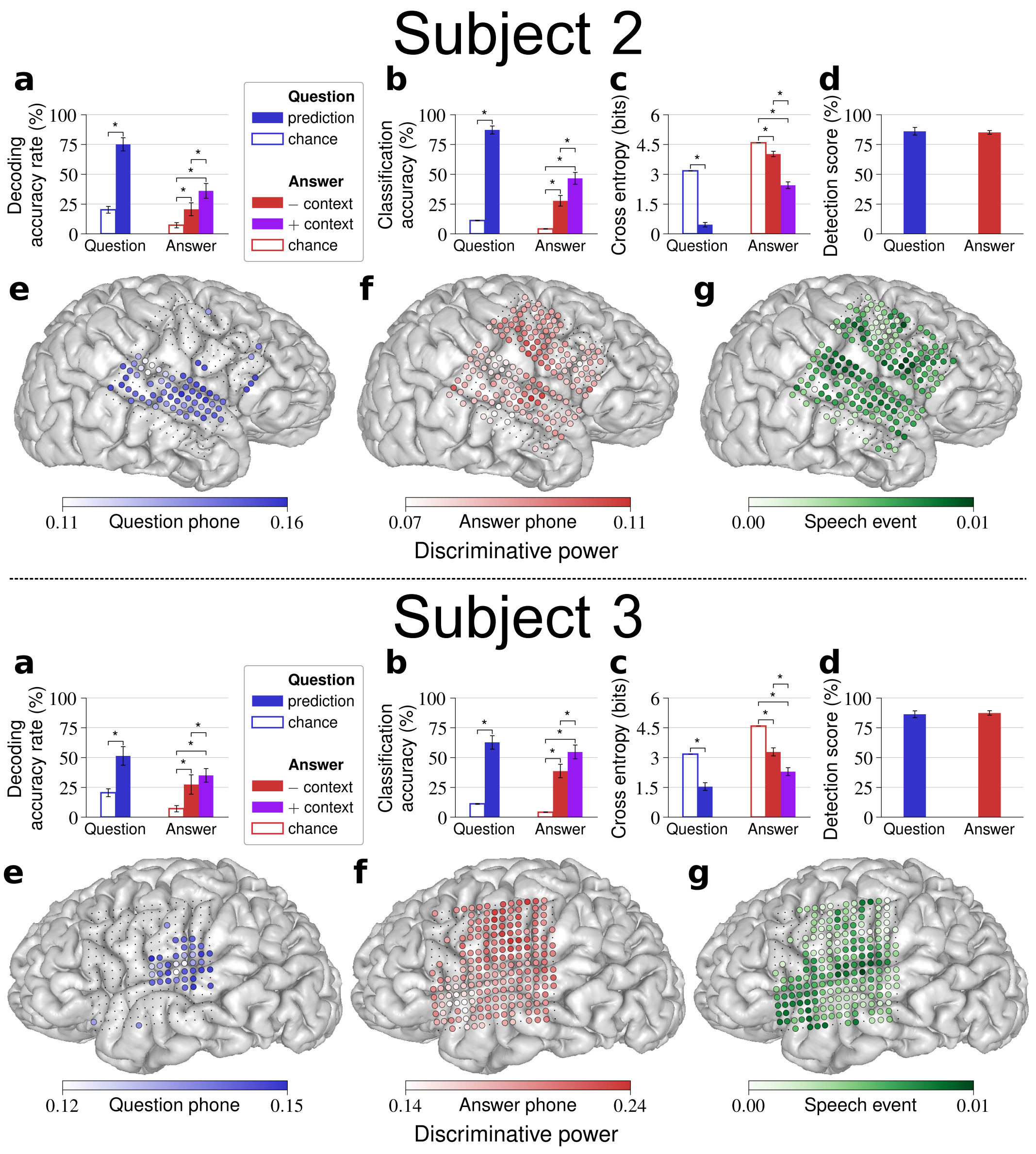
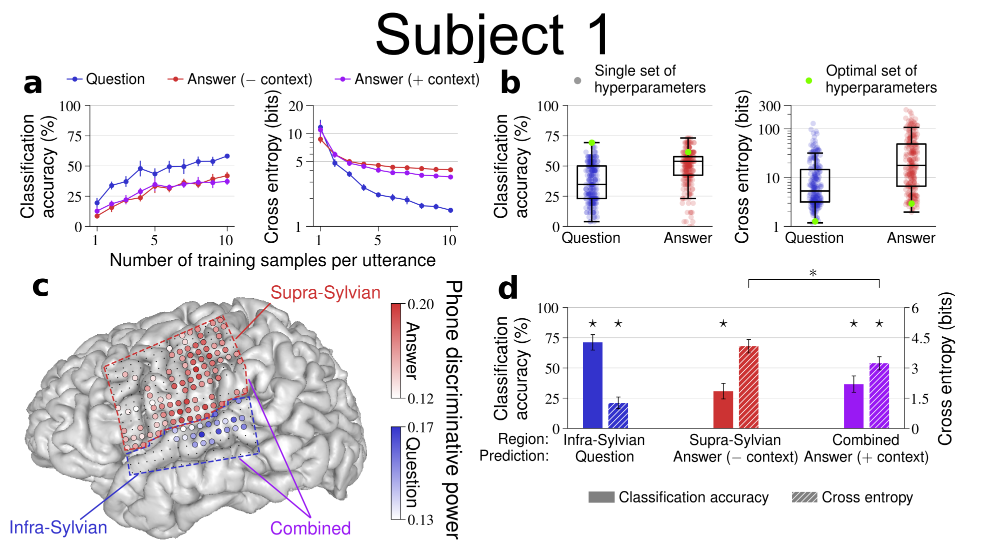


Figure 4.5: Decoding and classification results for questions, answers, and answers after integration of the decoded context for subjects 2 and 3. (**a**) Decoding accuracy rates, which measure the full performance of the system, are significantly above chance for questions and answers (with and without context). Accuracy is significantly higher with context compared to without context. (**b**) Classification accuracies (the percent of speech events in which the system correctly classified the utterance) mirror decoding accuracy rates. (**c**) Cross entropies for utterance classification exhibit similar significant differences (lower values indicate better performance). (**d**) Question and answer detection scores demonstrate near-ceiling performance of the speech detection model for both questions and answers. (**e**–**g**) MRI brain reconstructions with electrode locations and discriminative power for each electrode used by **e** question phone, **f** answer phone, and **g** speech event discriminative models. Electrodes that were not relevant for the current model are depicted as small black dots. In **a**–**d**, data are mean ± SEM. $^*P < 0.05$.



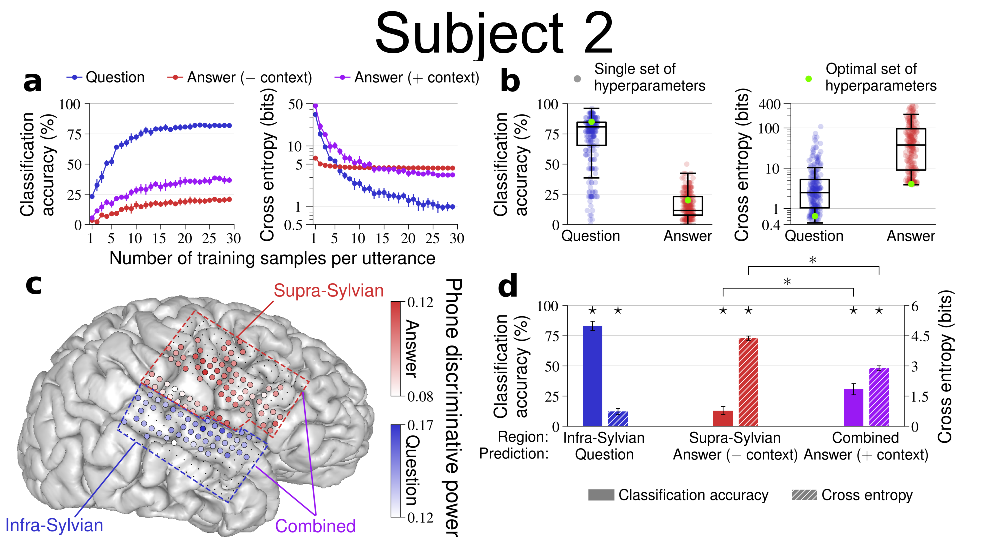


Figure 4.6: Data limitations, hyperparameter optimization, and functional anatomy of speech classification for subjects 1 and 2 (see Fig. 4.3 for full descriptions of each panel). (**a**) Classification performance as a function of the amount of training data. Performance does not seem to plateau for subject 1 (all prediction types) and for question classification with subject 2. (**b**) Variability in classification performance across hyperparameter optimization epochs for one test block. For each subject, hyperparameter selection has a large impact on performance, and each optimizer is able to choose hyperparameter values effectively. (**c**,**d**) Contributions of distinct brain regions on classification performance. **c** MRI reconstruction with electrode locations, relevant electrodes, and electrode phone discriminative powers. In this analysis, question classification models were fit and tested using infra-Sylvian electrodes, answer classification models were fit and tested using supra-Sylvian electrodes, and a combined model used likelihoods from both of these models to compute answer predictions with context. **d** Classification performance using these region-of-interest classification models. All models performed above chance except when assessing cross entropy for answer classifications with subject 1 ($^\star P < 0.05$), and performance was significantly improved when integrating context using the combined model except when assessing classification accuracy with subject 1 ($^* P < 0.05$). Overall, the classifiers exhibited reliable performance when limiting cortical coverage. In (**a**,**d**), data are mean ± SEM.

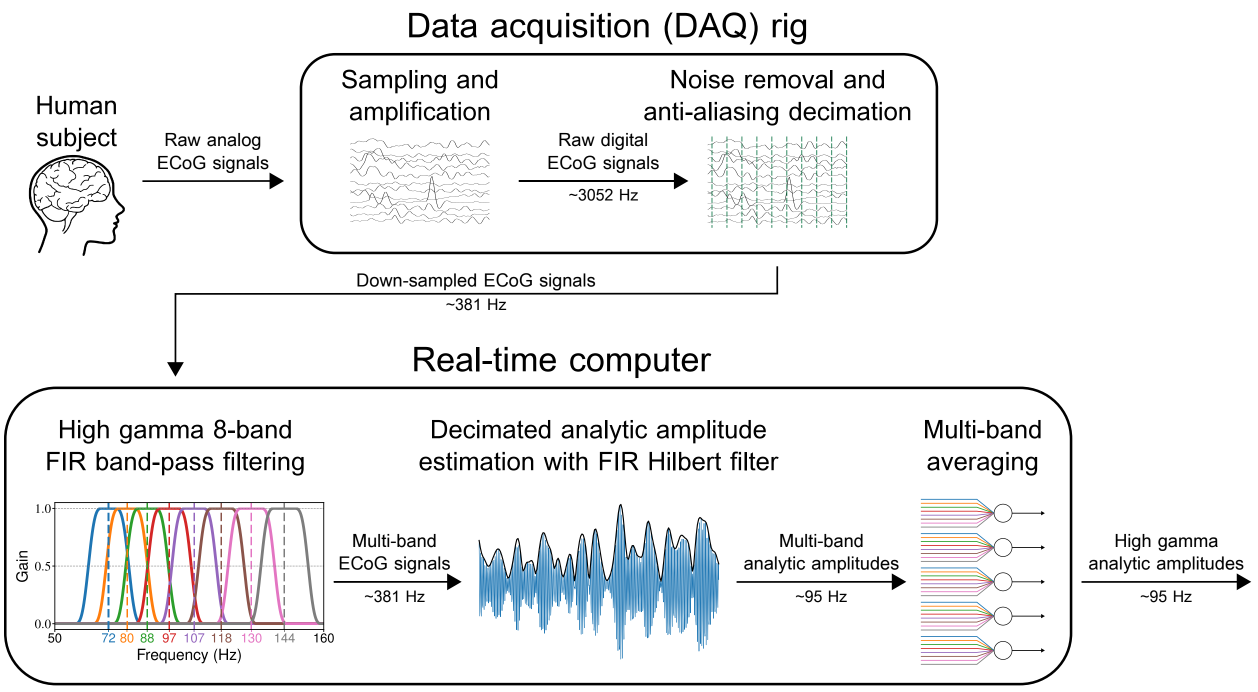

Data acquisition (DAQ) rig
Human subject
Raw analog ECoG signals
Sampling and amplification
Raw digital ECoG signals
~3052 Hz
Noise removal and anti-aliasing decimation
Down-sampled ECoG signals
~381 Hz
Real-time computer
High gamma 8-band FIR band-pass filtering
Gain
1.0
0.5
0.0
50
72
80
88
97
107
118
130
144
160
Frequency (Hz)
Multi-band ECoG signals
~381 Hz
Decimated analytic amplitude estimation with FIR Hilbert filter
Multi-band analytic amplitudes
~95 Hz
Multi-band averaging
High gamma analytic amplitudes
~95 Hz

Figure 4.7: Real-time neural signal preprocessing with the *rtNSR* system. In the DAQ rig, ECoG signals are sampled from the subject's brain at 3052 Hz, quantized, notch filtered at 60, 120, and 180 Hz, and decimated (with anti-aliasing) to 381 Hz. The resulting signals are streamed into the real-time computer and, within *rtNSR*, band-passed using eight FIR filters with center frequencies in the high gamma band (filter responses shown in the bottom-left plot). The analytic amplitude is then estimated for each of the eight band-passed signals for each channel at 95 Hz using an FIR filter designed to approximate the Hilbert transform. The analytic amplitudes for the eight bands associated with each channel are averaged to yield high gamma analytic amplitudes for each channel. All of the sampling rates given in this figure and caption were rounded to the nearest whole number.

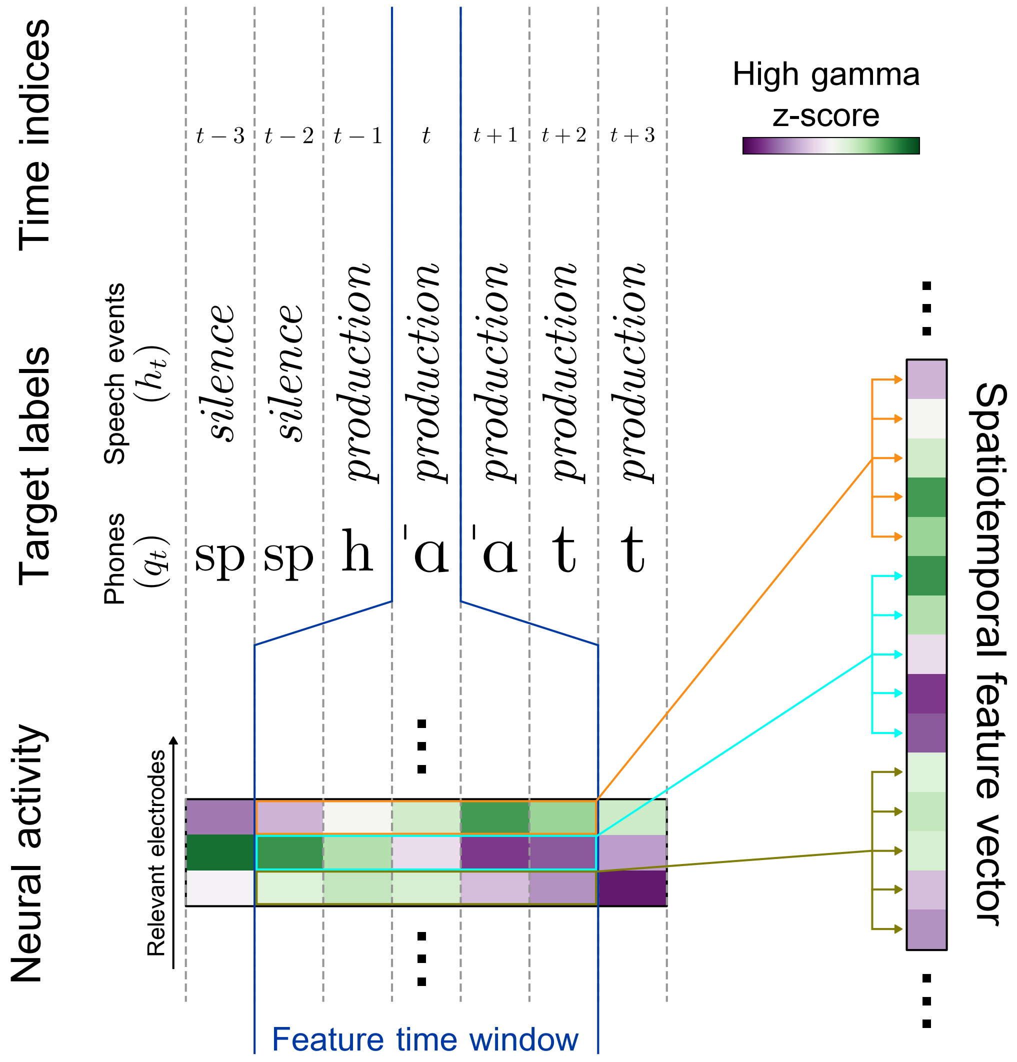


Figure 4.8: Spatiotemporal neural feature vectors and associated target labels during training of the speech detection and utterance classification models. In the depicted example, a subject starts to produce the answer utterance “Hot” (with phonetic transcription /h ˈɑ t/). The speech onset occurs at time index $t-1$. The phone labels q_t at each time point t are obtained from phonetic transcriptions. The speech event labels h_t, which are either *silence*, *perception*, or *production* at every time point, are determined directly from these phonetic transcriptions. The feature vector at time t contains the high gamma z-score value at every relevant electrode for every time point within some feature time window relative to t, representing the neural activity both spatially and temporally. The feature vector and target label for each time index are used to train the speech event probability and phone likelihood models. During testing, the neural feature vectors y_t are constructed in a similar fashion and used within the speech detection model to compute the speech event probabilities $p\left(h_t|y_t\right)$ and within the utterance classification models to compute the phone likelihoods $p\left(y_t|q_t\right)$.

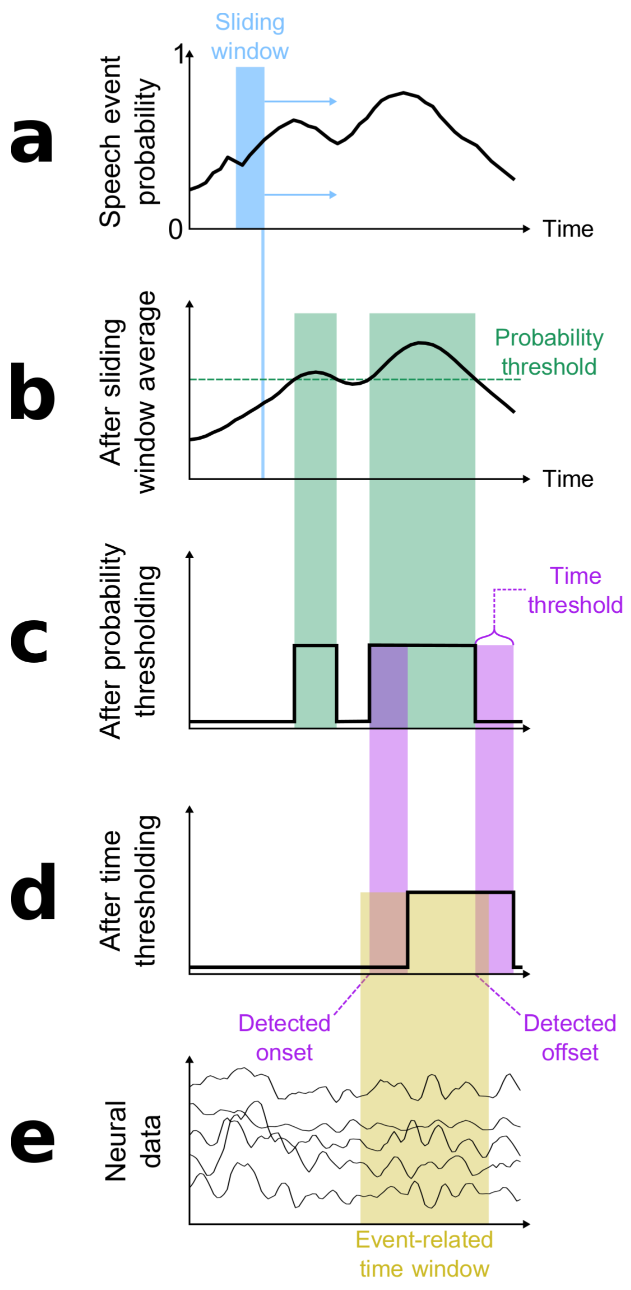


Figure 4.9: Speech event detection during real-time decoding. (**a**) First, speech event probabilities are computed by the speech event probability model for each time point. The plotted curve depicts example event probabilities for one of the utterance types (either questions or answers). (**b**) The speech event probabilities are smoothed using a sliding window average. (**c**) These smoothed probabilities are thresholded to either be 1 if the detection model suspects that a speech event is occurring or 0 otherwise. (**d**) The probability-thresholded binary values are then thresholded in time. Sometimes referred to as debouncing, this step prevents the binary values from switching rapidly between 0 and 1. A transition from 0 to 1 in the time-thresholded values signifies a speech onset, and a transition from 1 to 0 signifies a speech offset. (**e**) The neural data segmented by the detected speech onset and offset, including some time points before and after the detected window (controlled by hyperparameters), are then passed to the appropriate utterance classification model.

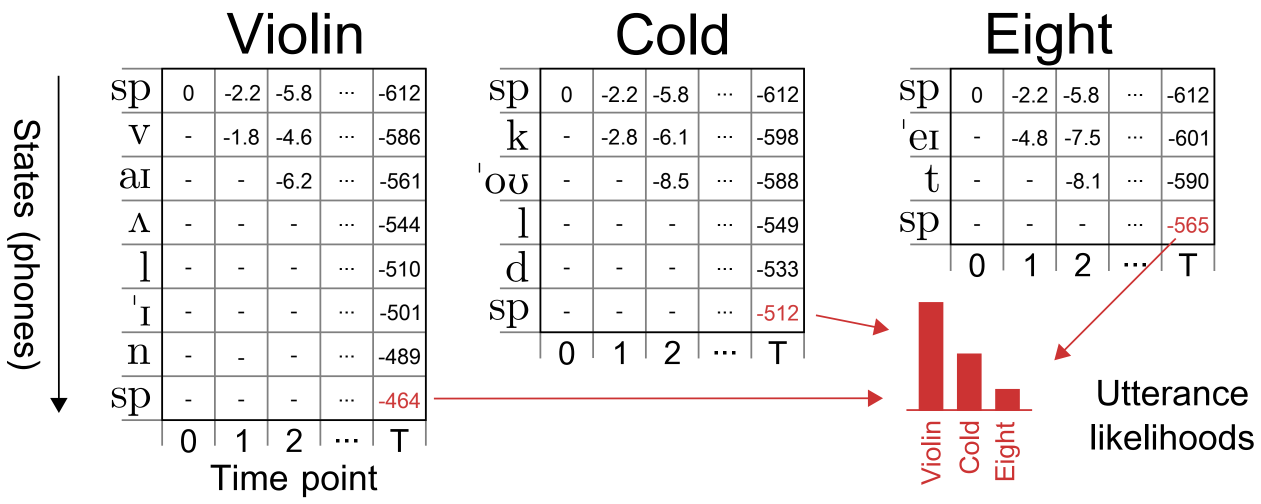

Violin
Cold
Eight
States (phones)
sp | 0 | -2.2 | -5.8 | ··· | -612
v | - | -1.8 | -4.6 | ··· | -586
aɪ | - | - | -6.2 | ··· | -561
ʌ | - | - | - | ··· | -544
l | - | - | - | ··· | -510
ˈɪ | - | - | - | ··· | -501
n | - | - | - | ··· | -489
sp | - | - | - | ··· | -464
0 1 2 ··· T
Time point
sp | 0 | -2.2 | -5.8 | ··· | -612
k | - | -2.8 | -6.1 | ··· | -598
ˈoʊ | - | - | -8.5 | ··· | -588
l | - | - | - | ··· | -549
d | - | - | - | ··· | -533
sp | - | - | - | ··· | -512
0 1 2 ··· T
sp | 0 | -2.2 | -5.8 | ··· | -612
ˈeɪ | - | -4.8 | -7.5 | ··· | -601
t | - | - | -8.1 | ··· | -590
sp | - | - | - | ··· | -565
0 1 2 ··· T
Violin
Cold
Eight
Utterance likelihoods

Figure 4.10: Use of Viterbi decoding in the utterance classification models to compute utterance log likelihoods. In this example, a classification model computes the likelihoods of the utterances “Violin” (/v aɪ ʌ l ˈɪ n/), “Cold” (/k ˈoʊ l d/), and “Eight” (/ˈeɪ t/). Each utterance is represented as an HMM with phones (obtained from the phonetic transcriptions) as hidden states and spatiotemporal neural feature vectors as observations. Each HMM is forced to have /sp/ as the first and last states. The transition matrix of each HMM is defined such that a phone state can only transition to itself or the next phone in the sequence (if it is not the last phone). Given feature vectors for time indices $t \in \{0, 1, \ldots, T\}$, Viterbi decoding is performed on each HMM, updating the values in the Viterbi trellis for each HMM (shown here as tables of log likelihoods) at each time index. The log likelihood of the most likely Viterbi path at the final state of each HMM (the value for the final /sp/ state at time T) is used as the log likelihood of that utterance. The classifier then smooths and normalizes these log likelihood values to obtain a final estimate for the utterance likelihoods associated with the provided feature vectors.

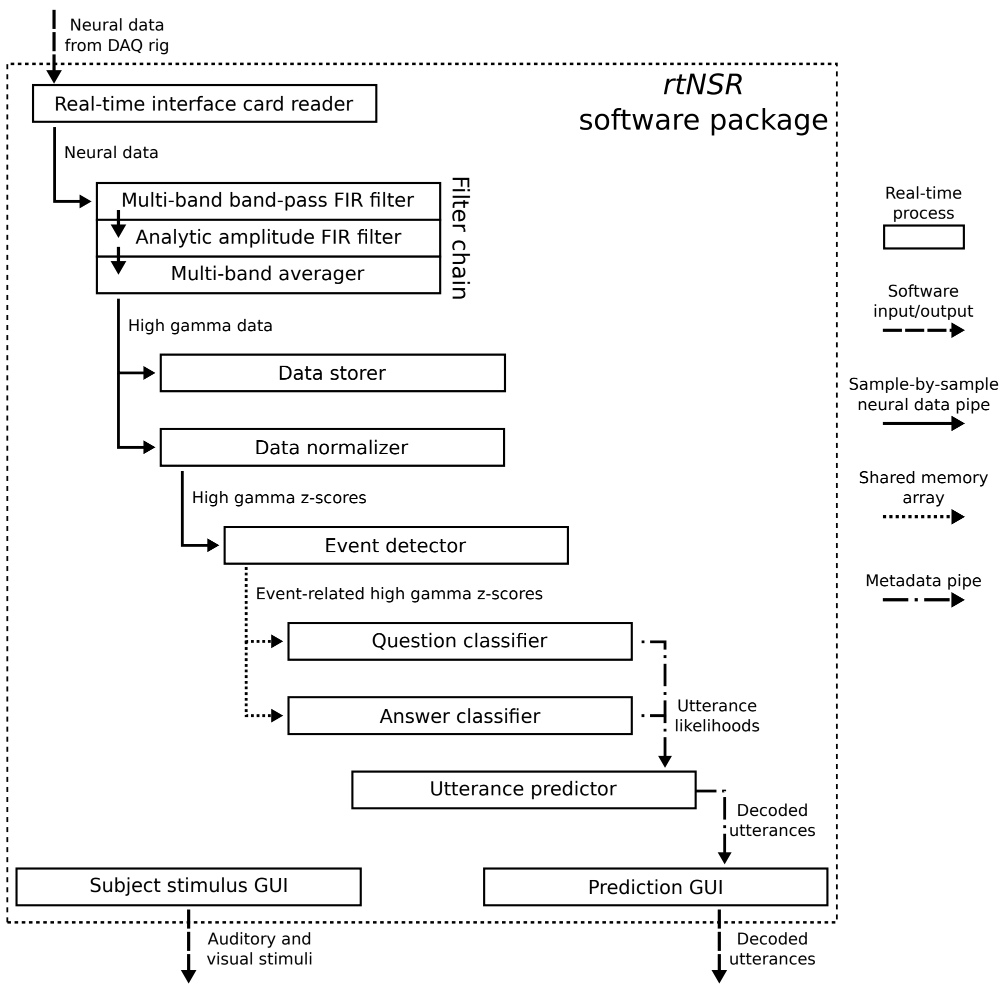


Figure 4.11: Schematic depiction of the *rtNSR* implementation used during real-time decoding. The solid rectangles represent real-time process classes and arrows represent the passing of information between the processes. The *Real-time interface card reader* process reads neural data acquired from the DAQ rig and streamed through the real-time interface card. The neural data are processed in a filter chain comprised of three processes: the *Multi-band band-pass FIR filter* process that band-passes the signals for each channel in eight different sub-bands in the high gamma band range (between 70–150 Hz), the *Analytic amplitude FIR filter* process that extracts the analytic amplitude for each band and each channel, and the *Multi-band averager* process that averages the analytic amplitude values across the bands for each channel to obtain that channel's high gamma activity. These high gamma signals are written to disk in the *Data storage* process (along with metadata from other processes, not depicted here) and normalized and clipped in the *Data normalizer* process. The normalized neural data are piped to the *Event detector* process, which analyzes the data at each time point to predict the onsets and offsets of speech events. When an event is detected, the high gamma z-scores are stored in a shared memory array that can be accessed by either the *Question classifier* or *Answer classifier* process to predict the utterance likelihoods associated with that event. The *Utterance predictor* process uses these likelihoods to update the answer priors and predict which question was heard or which answer was said by the subject. The *Prediction GUI* process displays the decoded utterances on a screen. Throughout the task, the *Subject stimulus GUI* process presents the auditory and visual stimuli to the subject.

References

W. H. Abdulla and N. K. Kasabov, 2001. Improving speech recognition performance through gender separation. *Artificial Neural Networks and Expert Systems*, pages 218–222.

American Congress of Rehabilitation Medicine, 1995. Recommendations for use of uniform nomenclature pertinent to patients with severe alterations in consciousness. *Archives of Physical Medicine and Rehabilitation*, 76(2):205–209. doi: 10.1016/S0003-9993(95)80031-X.

R. H. Baayen, D. J. Davidson, and D. M. Bates, 2008. Mixed-effects modeling with crossed random effects for subjects and items. *Journal of Memory and Language*, 59(4):390–412. doi: 10.1016/j.jml.2007.12.005.

T. Ball, M. Kern, I. Mutschler, A. Aertsen, and A. Schulze-Bonhage, 2009. Signal quality of simultaneously recorded invasive and non-invasive EEG. *NeuroImage*, 46(3):708–716. doi: 10.1016/j.neuroimage.2009.02.028.

D. J. Barr, R. Levy, C. Scheepers, and H. J. Tily, 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language*, 68 (3):255–278. doi: 10.1016/j.jml.2012.11.001.

D. Bates, M. Mächler, B. Bolker, and S. Walker, 2015. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48. doi: 10.18637/jss.v067.i01.

M. F. BenZeghiba, R. De Mori, O. Deroo, S. Dupont, T. Erbes, D. Jouvet, L. Fissore, P. Laface, A. Mertins, C. Ris, R. C. Rose, V. M. Tyagi, and C. J. Wellekens, oct 2007. Automatic speech recognition and speech variability: a review. *Speech Communication*, 49(10-11):763–786. doi: 10.1016/j.specom.2007.02.006.

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, 2011. Algorithms for Hyper-Parameter Optimization. *Advances in Neural Information Processing Systems (NIPS)*, pages 2546–2554. doi: 2012arXiv1206.2944S.

J. Bergstra, D. L. K. Yamins, and D. D. Cox, 2013. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. *Icml*, pages 115–123.

J. R. J. Binder, J. A. F. Bellgowan, T. A. Hammeke, P. Bellgowan, J. Springer, and J. N. Kaufman, 2000. Human temporal lobe activation by speech and nonspeech sounds. *Cerebral Cortex*, 10(5):512–528. doi: 10.1093/cercor/10.5.512.

A. W. Black, P. Taylor, and R. Caley, 1997. Festival Speech Synthesis System: System Documentation Edition 1.1. Technical report, Human Communciation Research Centre, University of Edinburgh, Scotland, UK.

D. F. Boatman, C. B. Hall, M. H. Goldstein, R. P. Lesser, and B. J. Gordon, mar 1997. Neuroperceptual differences in consonant and vowel discrimination: As revealed by direct cortical electrical interference. *Cortex*, 33(1):83–98. doi: 10.1016/S0010-9452(97)80006-8.

P. Boersma, 2001. Praat, a system for doing phonetics by computer. *Glot International*, 5 (9/10):341–345. doi: 10.1097/AUD.0b013e31821473f7.

K. E. Bouchard, N. Mesgarani, K. Johnson, and E. F. Chang, mar 2013. Functional organization of human sensorimotor cortex for speech articulation. *Nature*, 495(7441): 327–332. doi: 10.1038/nature11911.

J. F. Brugge, 1992. An overview of central auditory processing. In R. R. Fay and A. N. Popper, editors, *The mammalian auditory pathway: Neurophysiology*, chapter 1, pages 1–33. Springer-Verlag, New York, NY.

M.-A. Bruno, J. L. Bernheim, D. Ledoux, F. Pellas, A. Demertzi, and S. Laureys, 2011. A survey on self-assessed well-being in a cohort of chronic locked-in syndrome patients: happy majority, miserable minority. *BMJ open*, 1(1):e000039. doi: 10.1136/bmjopen-2010-000039.

M. Brysbaert and B. New, 2009. Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior Research Methods*, 41(4):977–990. doi: 10.3758/BRM.41.4.977.

D. V. Buonomano and W. Maass, feb 2009. State-dependent computations: spatiotemporal processing in cortical networks. *Nature reviews Neuroscience*, 10:113–125. doi: 10.1038/nrn2558.

R. T. Canolty, M. Soltani, S. S. Dalal, E. Edwards, N. F. Dronkers, S. S. Nagarajan, H. E. Kirsch, N. M. Barbaro, and R. T. Knight, 2007. Spatiotemporal dynamics of word processing in the human brain. *Frontiers in Neuroscience*, 1(1):185–196.

D. Carey, S. Krishnan, M. F. Callaghan, M. I. Sereno, and F. Dick, 2017. Functional and Quantitative MRI Mapping of Somatomotor Representations of Human Supralaryngeal Vocal Tract. *Cerebral cortex*, 27(1):265–278. doi: 10.1093/cercor/bhw393.

E. F. Chang, J. W. Rieger, K. Johnson, M. S. Berger, N. M. Barbaro, and R. T. Knight, nov 2010. Categorical speech representation in human superior temporal gyrus. *Nature neuroscience*, 13(11):1428–32. doi: 10.1038/nn.2641.

S.-y. Chang, E. Edwards, N. Morgan, D. Ellis, N. Mesgarani, and E. Chang, 2015. Phone

Recognition for Mixed Speech Signals : Comparison of Human Auditory Cortex and Machine Performance. TR-15-002.

J. Chartier, G. K. Anumanchipalli, K. Johnson, and E. F. Chang, 2018. Encoding of Articulatory Kinematic Trajectories in Human Speech Sensorimotor Cortex. *Neuron*, 98 (5):1042–1054.e4. doi: 10.1016/j.neuron.2018.04.031.

S. F. Chen and J. Goodman, 1998. An empirical study of smoothing techniques for language modeling. Technical Report August, Computer Science Group, Harvard University.

C. Cheung and E. F. Chang, aug 2012. Real-time, time-frequency mapping of event-related cortical activation. *Journal of neural engineering*, 9(4):046018. doi: 10.1088/1741-2560/9/4/046018.

N. Chomsky and M. Halle, 1968. *The sound pattern of English.* Harper & Row, New York.

E. S. Cibelli, M. K. Leonard, K. Johnson, and E. F. Chang, 2015. The influence of lexical statistics on temporal lobe cortical dynamics during spoken word listening. *Brain and Language*, 147:66–75. doi: http://dx.doi.org/10.1016/j.bandl.2015.05.005.

D. F. Conant, K. E. Bouchard, M. K. Leonard, and E. F. Chang, 2018. Human sensorimotor cortex control of directly-measured vocal tract movements during vowel production. *The Journal of Neuroscience*, 38(12):2382–17. doi: 10.1523/JNEUROSCI.2382-17.2018.

N. E. Crone, D. L. Miglioretti, B. Gordon, and R. P. Lesser, 1998. Functional mapping of human sensorimotor cortex with electrocorticographic spectral analysis. II. Event-related synchronization in the gamma band. *Brain*, 121(12):2301–2315. doi: 10.1093/brain/121.12.2301.

N. E. Crone, D. Boatman, B. Gordon, and L. Hao, apr 2001. Induced electrocorticographic gamma activity during auditory perception. *Clinical Neurophysiology*, 112(4):565–582. doi: 10.1016/S1388-2457(00)00545-9.

A. M. Dale, B. Fischl, and M. I. Sereno, 1999. Cortical surface-based analysis. *NeuroImage*, 9:179–194.

M. Davenport and S. Hannahs, 2010. *Introducing phonetics and phonology.* Routledge, New York, NY.

S. B. Davis and P. Mermelstein, 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366. doi: 10.1109/TASSP.1980.1163420.

B. Denby, T. Schultz, K. Honda, T. Hueber, J. M. Gilbert, and J. S. Brumberg, 2010. Silent speech interfaces. *Speech Communication*, 52(4):270–287. doi: 10.1016/j.specom.2009.08.002.

M. R. DeWeese, M. Wehr, and A. M. Zador, 2003. Binary spiking in auditory cortex. *Journal of Neuroscience*, 23(21):7940–7949. doi: 23/21/7940[pii].

B. K. Dichter, J. D. Breshears, M. K. Leonard, and E. F. Chang, 2018. The Control of Vocal Pitch in Human Laryngeal Motor Cortex. *Cell*, 174(1):21–31.e9. doi: 10.1016/j.cell.2018.05.016.

J. L. Elman, 1990. Finding structure in time. *Cognitive science*, 14(2):179–211. doi: 10.1207/s15516709cog1402_1.

C. T. Engineer, C. A. Perez, Y. H. Chen, R. S. Carraway, A. C. Reed, A. Shetake, V. Jakkamsetti, K. Q. Chang, and M. P. Kilgard, 2008. Cortical activity patterns predict speech discrimination ability. *Nature Neuroscience*, 11(5):603–608. doi: 10.1038/nn.2109. Cortical.

R. A. Fisher, 1932. *Statistical methods for research workers.* Oliver & Boyd, Edinburgh, Scotland, 4th editio edition.

J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, N. Dahlgren, and V. Zue, 1993. TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1. *Linguistic Data Consortium*, page 1.

B. Gold, N. Morgan, and D. Ellis, 2011. *Speech and audio signal processing: processing and perception of speech and music*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2nd edition.

A. Graves, A.-r. Mohamed, and G. Hinton, 2013. Speech recognition with deep recurrent neural networks. In *International Conference on Acoustics, Speech, and Signal Processing*, number 3, pages 6645–6649. doi: 10.1109/ICASSP.2013.6638947.

R. Haeb-Umbach and H. J. Ney, 1992. Linear discriminant analysis for improved large vocabulary continuous speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 13–16. doi: 10.1109/ICASSP.1992.225984.

L. S. Hamilton, D. L. Chang, M. B. Lee, and E. F. Chang, 2017. Semi-automated Anatomical Labeling and Inter-subject Warping of High-Density Intracranial Recording Electrodes in Electrocorticography. *Frontiers in Neuroinformatics*, 11(October):62. doi: 10.3389/fninf.2017.00062.

L. S. Hamilton, E. Edwards, and E. F. Chang, 2018. A Spatial Map of Onset and Sustained Responses to Speech in the Human Superior Temporal Gyrus. *Current Biology*, 28(12): 1860–1871.e4. doi: 10.1016/j.cub.2018.04.033.

T. Hastie, R. Tibshirani, and J. Friedman, 2009. *The elements of statistical learning: Data mining, inference, and prediction*. Springer, New York, NY, 2nd edition.

C. Herff, D. Heger, A. de Pesters, D. Telaar, P. Brunner, G. Schalk, and T. Schultz, 2015. Brain-to-text: decoding spoken phrases from phone representations in the brain. *Frontiers in Neuroscience*, 9(June):1–11. doi: 10.3389/fnins.2015.00217.

D. Hermes, K. J. Miller, H. J. Noordmans, M. J. Vansteensel, and N. F. Ramsey, 2010. Automated electrocorticographic electrode localization on individually rendered brain surfaces. *Neuroscience Methods*, 185(2):293–298.

G. Hickok and D. Poeppel, 2007. The cortical organization of speech processing. *Nature Reviews Neuroscience*, 8(May):393–402. doi: 10.1038/nrn2113.

G. Hinton, L. Deng, D. Yu, G. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97. doi: 10.1109/MSP.2012.2205597.

G. Hotson, D. P. McMullen, M. S. Fifer, M. S. Johannes, K. D. Katyal, M. P. Para, R. Armiger, W. S. Anderson, N. V. Thakor, B. A. Wester, and N. E. Crone, 2016. Individual finger control of a modular prosthetic limb using high-density electrocorticography in a human subject. *Journal of Neural Engineering*, 13(2):026017. doi: 10.1088/1741-2560/13/2/026017.

X. Huang, A. Acero, and H.-W. Hon, 2001. *Spoken language processing: a guide to theory, algorithm and system development*. Prentice-Hall, Upper Saddle River, New Jersey, 1st edition.

A. G. Huth, S. Nishimoto, A. T. Vu, and J. L. Gallant, 2012. A Continuous Semantic Space Describes the Representation of Thousands of Object and Action Categories across the Human Brain. *Neuron*, 76(6):1210–1224. doi: 10.1016/j.neuron.2012.10.014.

T. F. Jaeger, 2008. Categorical data analysis: Away from ANOVAs (transformation or not) and towards logit mixed models. *Journal of Memory and Language*, 59(4):434–446. doi: 10.1016/j.jml.2007.11.007.

D. Jurafsky and J. H. Martin, 2009. *Speech and language processing: an introduction to* *natural language processing, computational linguistics, and speech recognition.* Pearson Education, Inc., Upper Saddle River, New Jersey, 2nd edition.

V. G. Kanas, I. Mporas, H. L. Benz, K. N. Sgarbas, A. Bezerianos, and N. E. Crone, 2014. Real-time voice activity detection for ECoG-based speech brain machine interfaces. *International Conference on Digital Signal Processing, DSP*, 2014-Janua(August):862–865. doi: 10.1109/ICDSP.2014.6900790.

S. Kellis, K. Miller, K. Thomson, R. Brown, P. House, and B. Greger, 2010. Decoding spoken words using local field potentials recorded from the cortical surface. *Journal of neural engineering*, 7(5):056007. doi: 10.1088/1741-2560/7/5/056007.

B. Khalighinejad, T. Nagamine, A. Mehta, and N. Mesgarani, 2017. NAPLib: An open source toolbox for real-time and offline Neural Acoustic Processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 846–850. doi: 10.1109/ICASSP.2017.7952275.

R. Kneser and H. Ney, 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184. doi: 10.1109/ICASSP.1995.479394.

C. Kurian, 2014. A review on technological development of automatic speech recognition. *International Journal of Soft Computing and Engineering*, 4(4):80–86.

M. Lam, 1988. Software pipelining: an effective scheduling technique for VLIW machines. *ACM SIGPLAN Notices*, 23(7):318–328. doi: 10.1145/960116.54022.

S. Laureys, F. Pellas, P. Van Eeckhout, S. Ghorbel, C. Schnakers, F. Perrin, J. Berré, M. E. Faymonville, K. H. Pantke, F. Damas, M. Lamy, G. Moonen, and S. Goldman, jan 2005. The locked-in syndrome: What is it like to be conscious but paralyzed and voiceless? *Progress in Brain Research*, 150(5):495–511. doi: 10.1016/S0079-6123(05)50034-7.

O. Ledoit and M. Wolf, 2004. Honey, I Shrunk the Sample Covariance Matrix. *The Journal of Portfolio Management*, 30(4):110–119. doi: 10.3905/jpm.2004.110.

M. K. Leonard, K. E. Bouchard, C. Tang, and E. F. Chang, 2015. Dynamic encoding of speech sequence probability in human temporal cortex. *Journal of Neuroscience*, 35(18): 7203–7214. doi: 10.1523/JNEUROSCI.4100-14.2015.

M. K. Leonard, M. O. Baud, M. J. Sjerps, and E. F. Chang, 2016. Perceptual restoration of masked speech in human cortex. *Nature Communications*, 7:13619. doi: 10.1038/ncomms13619.

E. C. Leuthardt, K. J. Miller, G. Schalk, R. P. Rao, and J. G. Ojemann, 2006. Electrocorticography-based brain computer interface - The seattle experience. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):194–198. doi: 10.1109/TNSRE.2006.875536.

E. C. Leuthardt, C. Gaona, M. Sharma, N. Szrama, J. Roland, Z. Freudenberg, J. Solis, J. Breshears, and G. Schalk, 2011. Using the electrocorticographic speech network to control a brain-computer interface in humans. *Journal of Neural Engineering*, 8(3):036004. doi: 10.1088/1741-2560/8/3/036004.

G. J. Lidstone, 1920. Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities. In *Transactions of the Faculty Actuaries*, volume 8, pages 182–192.

J. F. Linden and C. E. Schreiner, 2003. Columnar transformations in auditory cortex? A comparison to visual and somatosensory cortices. *Cerebral cortex (New York, N.Y. : 1991)*, 13(1):83–89. doi: 10.1093/cercor/13.1.83.

F. Lotte, J. S. Brumberg, P. Brunner, A. Gunduz, A. L. Ritaccio, C. Guan, and G. Schalk, 2015. Electrocorticographic representations of segmental features in continuous speech. *Frontiers in Human Neuroscience*, 09(February):1–13. doi: 10.3389/fnhum.2015.00097.

K. A. Ludwig, R. M. Miriani, N. B. Langhals, M. D. Joseph, D. J. Anderson, and D. R. Kipke, mar 2009. Using a common average reference to improve cortical neuron recordings from microelectrode arrays. *Journal of neurophysiology*, 101(3):1679–89. doi: 10.1152/jn.90989.2008.

H. Luo and D. Poeppel, 2007. Phase patterns of neuronal responses reliably discriminate speech in human auditory cortex. *Neuron*, 54(6):1001–1010. doi: 10.1016/j.neuron.2007.06.004.

B. O. Mainsah, L. M. Collins, K. a. Colwell, E. W. Sellers, D. B. Ryan, K. Caves, and C. S. Throckmorton, 2015. Increasing BCI communication rates with dynamic stopping towards more practical use: An ALS study. *Journal of Neural Engineering*, 12(1):016013. doi: 10.1088/1741-2560/12/1/016013.

S. L. Marple and S. Lawrence Marple, 1999. Computing the discrete-time analytic signal via fft. *IEEE Transactions on Signal Processing*, 47(9):2600–2603. doi: 10.1109/78.782222.

S. Martin, 2017. *Understanding and Decoding Imagined Speech using Electrocorticographic Recordings in Humans*. PhD thesis, EPFL.

S. Martin, P. Brunner, C. Holdgraf, H.-J. Heinze, N. E. Crone, J. Rieger, G. Schalk, R. T. Knight, and B. N. Pasley, 2014. Decoding spectrotemporal features of overt and covert speech from the human cortex. *Frontiers in neuroengineering*, 7(May):14. doi: 10.3389/fneng.2014.00014.

S. S. Martin, P. Brunner, I. Iturrate, J. d. R. Millán, G. Schalk, R. T. Knight, and B. N. Pasley, 2016. Word pair classification during imagined speech using direct brain recordings. *Scientific Reports*, 6(1):25803. doi: 10.1038/srep25803.

Q. McNemar, 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157. doi: 10.1007/BF02295996.

N. Mesgarani and E. F. Chang, may 2012. Selective cortical representation of attended speaker in multi-talker speech perception. *Nature*, 485(7397):233–6. doi: 10.1038/nature11020.

N. Mesgarani, C. Cheung, K. Johnson, and E. F. Chang, feb 2014. Phonetic feature encoding in human superior temporal gyrus. *Science*, 343(6174):1006–1010. doi: 10.1126/science.1245994.

T. M. Mitchell, S. V. Shinkareva, A. Carlson, K.-M. M. Chang, V. L. Malave, R. A. Mason, and M. A. Just, may 2008. Predicting human brain activity associated with the meanings of nouns. *Science*, 320(5880):1191–1195. doi: 10.1126/science.1152876.

M. Moerel, F. De Martino, and E. Formisano, 2014. An anatomical and functional topography of human auditory cortical areas. *Frontiers in Neuroscience*, 8(8 JUL):1–14. doi: 10.3389/fnins.2014.00225.

D. A. Moses, N. Mesgarani, M. K. Leonard, and E. F. Chang, 2016. Neural speech recognition: Continuous phoneme decoding using spatiotemporal representations of human cortical activity. *Journal of Neural Engineering*, 13(5):056004. doi: 10.1088/1741-2560/13/5/056004.

D. A. Moses, M. K. Leonard, and E. F. Chang, 2018. Real-time classification of auditory sentences using evoked cortical activity in humans. *Journal of Neural Engineering*, 15(3). doi: 10.1088/1741-2552/aaab6f.

E. M. Mugler, J. L. Patton, R. D. Flint, Z. a. Wright, S. U. Schuele, J. Rosenow, J. J. Shih, D. J. Krusienski, and M. W. Slutzky, 2014. Direct classification of all American English phonemes using signals from functional speech motor cortex. *Journal of neural engineering*, 11(3):035015. doi: 10.1088/1741-2560/11/3/035015.

T. Ogawa, J. Riera, T. Goto, A. Sumiyoshi, H. Nonaka, K. Jerbi, O. Bertrand, and R. Kawashima, 2011. Large-scale heterogeneous representation of sound attributes in rat primary auditory cortex: from unit activity to population dynamics. *Journal of Neuroscience*, 31(41):14639–14653. doi: 10.1523/JNEUROSCI.0086-11.2011.

H. Okamoto and R. Kakigi, 2014. Neural adaptation to silence in the human auditory cortex: a magnetoencephalographic study. *Brain and Behavior*, 4(6):858–866. doi: 10.1002/brb3.290.

A. V. Oppenheim, R. W. Schafer, and J. R. Buck, 1999. *Discrete-time signal processing.* Prentice-Hall, Upper Saddle River, New Jersey, 2nd edition.

T. W. Parks and J. H. McClellan, 1972. Chebyshev Approximation for Nonrecursive Digital Filters with Linear Phase. *IEEE Transactions on Circuit Theory*, 19(2):189–194. doi: 10.1109/TCT.1972.1083419.

B. N. Pasley, S. V. David, N. Mesgarani, A. Flinker, S. A. Shamma, N. E. Crone, R. T. Knight, and E. F. Chang, jan 2012. Reconstructing speech from human auditory cortex. *PLoS Biology*, 10(1):e1001251. doi: 10.1371/journal.pbio.1001251.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg, 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12:2825–2830. doi: 10.1007/s13398-014-0173-7.2.

X. Pei, D. L. Barbour, E. C. Leuthardt, and G. Schalk, 2011. Decoding vowels and consonants in spoken and imagined words using electrocorticographic signals in humans. *Journal of Neural Engineering*, 8(4):046028. doi: 10.1088/1741-2560/8/4/046028.

Python Software Foundation. Python Language Reference, 2010.

Python Software Foundation. Python programming language, 2016.

R Core Team. R: A language and environment for statistical computing, 2015.

L. R. Rabiner and B.-H. Juang, 1993. *Fundamentals of speech recognition.* Prentice-Hall, Upper Saddle River, New Jersey.

J. P. Rauschecker and S. K. Scott, jun 2009. Maps and streams in the auditory cortex: nonhuman primates illuminate human speech processing. *Nature neuroscience*, 12(6): 718–724. doi: 10.1038/nn.2331.

S. Ray, N. E. Crone, E. Niebur, P. J. Franaszczuk, and S. S. Hsiao, 2008. Neural correlates of high-gamma oscillations (60-200 Hz) in macaque local field potentials and their potential implications in electrocorticography. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 28(45):11526–11536. doi: 10.1523/JNEUROSCI.2848-08.2008.

D. E. T. Romero and G. Jovanovic, 2012. Digital FIR Hilbert Transformers: Fundamentals and Efficient Design Methods. In *MATLAB - A Fundamental Tool for Scientific Computing and Engineering Applications - Volume 1*, chapter 19, pages 445–482. doi: 10.5772/46451.

M.-C. Rousseau, K. Baumstarck, M. Alessandrini, V. Blandin, T. Billette de Villemeur, and P. Auquier, 2015. Quality of life in patients with locked-in syndrome: Evolution over a 6-year period. *Orphanet journal of rare diseases*, 10:88. doi: 10.1186/s13023-015-0304-z.

E. W. Sellers, D. B. Ryan, and C. K. Hauser, oct 2014. Noninvasive brain-computer interface enables communication after brainstem stroke. *Science translational medicine*, 6(257): 257re7. doi: 10.1126/scitranslmed.3007801.

M. Spüler, W. Rosenstiel, and M. Bogdan, 2012. Online Adaptation of a c-VEP Brain-Computer Interface(BCI) Based on Error-Related Potentials and Unsupervised Learning. *PLoS ONE*, 7(12). doi: 10.1371/journal.pone.0051077.

M. Steinschneider, K. V. Nourski, H. Kawasaki, H. Oya, J. F. Brugge, and M. a. Howard,

2011. Intracranial study of speech-elicited activity on the human posterolateral superior temporal gyrus. *Cerebral Cortex*, 21(10):2332–2347. doi: 10.1093/cercor/bhr014.

C. Tang, L. S. Hamilton, and E. F. Chang, 2017. Intonational speech prosody encoding in the human auditory cortex. *Science (New York, N.Y.)*, 357(August):797–801. doi: 10.1126/science.aam8577.

The MathWorks Inc. MATLAB, version 8.1.0, 2013.

X. Tian and D. Poeppel, 2010. Mental imagery of speech and movement implicates the dynamics of internal forward models. *Frontiers in Psychology*, 1(OCT):1–23. doi: 10.3389/fpsyg.2010.00166.

H. Tiitinen, I. Miettinen, P. Alku, and P. J. C. May, 2012. Transient and sustained cortical activity elicited by connected speech of varying intelligibility. *BMC neuroscience*, 13(1): 157. doi: 10.1186/1471-2202-13-157.

M. J. Vansteensel, E. G. Pels, M. G. Bleichner, M. P. Branco, T. Denison, Z. V. Freudenburg, P. Gosselaar, S. Leinders, T. H. Ottens, M. A. Van Den Boom, P. C. Van Rijen, E. J. Aarnoutse, and N. F. Ramsey, 2016. Fully Implanted Brain–Computer Interface in a Locked-In Patient with ALS. *New England Journal of Medicine*, 375(21):NEJMoa1608085. doi: 10.1056/NEJMoa1608085.

A. J. Viterbi, 1967. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269. doi: 10.1109/TIT.1967.1054010.

R. Weide. The {CMU} pronunciation dictionary, release 0.7a, 2014.

M. Yang, S. A. Sheth, C. A. Schevon, G. M. Mckhann Ii, N. Mesgarani, G. M. M. Ii, and N. Mesgarani, 2015. Speech reconstruction from human auditory cortex with deep neural networks. In *Sixteenth Annual Conference of the International Speech Communication Association*, pages 1121–1125.

S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, 2002. *The HTK book*, volume 3. doi: 10.1.1.124.3972.

J. Yuan and M. Liberman, 2008. Speaker identification on the SCOTUS corpus. *The Journal of the Acoustical Society of America*, 123(5):3878–3878. doi: 10.1121/1.2935783.