Master Thesis

Encoder-decoder transformer for speech neuroprosthesis trained on multiple sEEG datasets

Michał Kalbarczyk

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Data Science for Decision Making
at the Department of Advanced Computing Sciences
of the Maastricht University

Thesis Committee:

Dr. ir. K. Driessens
Dr. Ch. Herff
Dr. P. Bonizzi

Maastricht University
Faculty of Science and Engineering
Department of Advanced Computing Sciences

October, 2024

Abstract

Speech neuroprosthesis, an application of Brain-Computer Interface (BCI) technology, aims to restore communication for individuals with speech impairments by synthesizing speech directly from brain activity through machine learning models. Typically, brain activity is collected via invasive methods, resulting in patient-specific datasets due to variability in electrode placement and count, as well as individual neurophysiological differences. Consequently, developing machine learning models that can efficiently and accurately decode speech neuroprosthesis signals is challenging, given limited data availability and the need for patient-specific models.

This research addresses these challenges by using autoencoders to project stereoelectroencephalography (sEEG) neural data from different patients into a common representational subspace of consistent dimensions. This shared representation is then used as input into a cross-patient encoder-decoder transformer model, which features a unique encoder per patient and a shared decoder across patients, enabling joint training on data from multiple individuals. Additionally, individual encoder-decoder transformer models for each patient are tested, as this specific transformer architecture has seen minimal application in speech neuroprosthesis. A pre-trained version of the model is also examined to determine if new patients can benefit from previously trained decoders of the transformer model.

Experimental results indicate that the cross-patient model does not gain significant benefit from increased data volume, regardless of autoencoder use. However, findings reveal the potential of encoder-decoder transformer models for speech neuroprosthesis, as the model successfully captured unique vocal characteristics for certain patients. Additionally, the experiments demonstrate the feasibility of pre-trained models by reusing a decoder for new patients, showing improved predictive performance compared to models trained from scratch.

Acknowledgments

First and foremost, I would like to express my gratitude to my Master's Thesis supervisors, Kurt Driessens and Christian Herff, for their unwavering support and patience. Thanks to their mentorship, I have grown as a researcher, gained valuable knowledge, and expanded my skill set. Their insightful discussions provided me with a fresh perspective and deepened my appreciation for various machine learning concepts. I am also thankful for the opportunity they created for me; although this journey has been challenging, I have thoroughly enjoyed working on this fascinating topic at the boundary of Artificial intelligence and the human brain.

Next, I would like to thank my colleagues at the Neural Interfacing Lab, particularly Joaquín Amigó Vega, for generously sharing and explaining his invaluable code with me. Additionally, I want to extend my gratitude to Paweł Maka, a PhD student in the DACS department, for his openness and willingness to assist me in navigating the remote server.

I also want to thank my parents for putting me in this privileged position, where I can continue my higher education and focus on it entirely.

Last but certainly not least, I want to express my deepest appreciation to my dearest girlfriend, Malwina. Her mental and physical support throughout the entire process of writing this Master's Thesis has been invaluable. Her encouragement filled me with confidence and determination, helping me overcome many challenges. Malwina has been my voice of reason, reminding me to avoid overworking myself and preventing burnout. She has emphasized the importance of self-care, maintaining a positive outlook, and focusing on the positive aspects of this journey.

Contents

Chapter 1

Introduction

Speech is one of the most fundamental human abilities, and losing even partial control over it can drastically reduce quality of life. Neurological disorders such as stroke or spinal cord injury can significantly impair communication by disrupting brain function or severing connections to the body, making verbal communication challenging or even impossible[1].

Augmentative and Alternative Communication (AAC) devices offer a potential solution by enabling users to operate communication tools using voluntary muscle movement[2]. For instance, AAC devices often feature modified screens or keyboards, allowing for easier control through eye or head movements. However, these devices have low word-per-minute rates, can be cumbersome and frustrating to use, and rely heavily on motor functions. For example, individuals with locked-in syndrome or amyotrophic lateral sclerosis struggle to use AAC devices, as not only their verbal but also motor functions are limited.

Brain-Computer Interface The Brain-Computer Interface (BCI) is an emerging field offering technology that does not require motor input to operate. It allows for establishing a direct communication link between the human brain and external devices[1]. These systems are designed to read and process brain activity to achieve desired goals and are primarily used in the medical domain as assistive devices that replace lost functions, or as rehabilitative tools that aim to restore cognitive abilities[3]. The core framework involves brain signal measurement devices, such as electroencephalogram (EEG), electrocorticography (ECoG), or stereoelectroencephalography (sEEG), which record brain activity. The recorded signals are then processed and used as input for decoding algorithms to produce application-specific outputs.

Speech Neuroprosthesis

Among various BCI approaches to restore communication abilities, one of the most promising is speech neuroprosthesis[4]. This application decodes brain activity associated with speech production to synthesize speech or text directly from neural data, enabling higher word-per-minute rates and facilitating more natural-sounding speech, which helps users express themselves more effectively. Although the field is progressing rapidly, significant challenges remain in making this technology accessible and reliable for individuals with severe speech impairments.

Speech Neuroprosthesis - general framework

Speech production is a complex process involving multiple features—such as articulatory, acoustic, and semantic representations, distributed across various brain regions [5]. Therefore, invasive data acquisition methods, such as ECoG and sEEG, are preferred over non-invasive techniques due to their higher spatial and temporal resolution and higher signal-to-noise ratio[4][6][7]. In a real-time speech neuroprosthesis pipeline[8][9], neural signals are recorded in short time windows, after which they are processed and passed to the main decoding algorithm. This algorithm processes the neural input to predict a sequence of intermediate audio representations (e.g., mel-spectrograms or speech units) rather than directly producing an audio waveform. These intermediate representations are then fed into a pre-trained vocoder, which synthesizes the final audio output. To achieve real-time processing, each step in this pipeline must operate on a small time scale, measured in milliseconds. Therefore, one of the challenges is that decoding model architectures must be efficient—complexity must be minimized to avoid delays, yet the models must be powerful enough to generate predictions allowing for accurate and intelligible speech synthesis.

In this Master's Thesis, however, a different issue is addressed. While various decoding algorithms and pipelines exist (see Section 2.2.1), they all face a fundamental challenge: as with any other machine learning algorithms the decoding models are highly dependent on data. This reliance introduces three primary challenges in developing speech neuroprosthesis and other BCI applications.

Speech Neuroprosthesis Data - challenges and possible solutions First, models are created for individual patients. The human brain is a highly complex structural and functional network composed of billions of interconnected neurons[10]. These neurons connect in characteristic ways, from small-scale micro-columns to large-scale functional systems, creating unique neural patterns shaped by factors like age, gender, disease, and environment[11]. This variability complicates the development of generalized machine learning models, as each patient's brain activity is described by a different underlying distribution, making cross-patient modeling a significant challenge. This signal uniqueness, also means that developing decoding models for individuals who are unable to produce speech presents additional challenges, as these patients cannot provide personal speech data for model training.

Second, collecting large amounts of data in the neuroscience domain is a challenging task. Volunteers must be found and must undergo lengthy data acquisition procedures. For invasive methods, which provide higher-quality data, the challenges are even greater due to the associated risks of the procedures and the limited availability of patients willing or able to participate[12]. This limited data availability, combined with the difficulties of cross-patient modeling, makes it harder to effectively use data-hungry deep learning models that are often required to achieve state-of-the-art performance.

Finally, the creation of generalizable models is further complicated by the variability inherent in certain data acquisition methods. Non-invasive techniques, such as EEG, can offer consistency in channel count and the location of covered brain areas, at the cost of lower data quality[12]. In contrast, as mentioned above, invasive methods provide higher-quality data but introduce greater variability across patient datasets due to differences introduced by the data acquisition process. For example, sEEG is commonly used to identify seizure zones in patients with epilepsy[7]. During this procedure, electrodes are implanted in the brain, but the number and placement of these electrodes vary from patient to patient. This leads to differences between datasets, as each

dataset has different dimensionality and captures information from distinct regions of the brain.

Overall, this data-related issue slows down the development of the field by putting constraints on the developed models and their accessibility to people in need. Complex machine learning models require huge amounts of data, and without them, their development is hindered.

While those problems might be difficult, they are not impossible to overcome, and certain studies have tried to provide a solution. There are two main directions for solving those challenges: either by focusing on the data or the model.

On the data side, different patient datasets can be projected into a shared representational space through an alignment process, enabling the generalization of brain waves across patients by creating embeddings. When comparing two datasets, A and B, their feature sets X_A and X_B must be consistent in both size and characteristics. This means that the number of features should be the same, and each feature should represent the same aspect of the data in both datasets. Methods such as manifold alignment specialize in creating a shared representational space for different datasets[13].

On the model side, cross-patient models aim to achieve consistent performance across datasets that are described by different distributions. These models, incorporate in some way an aligning mechanism in their decoding pipeline. This can be done through distance-based learning or by carefully designing a suitable network architecture that can work an arbitrary number of channels. These models often rely on autoencoders, which compress input data into a latent space where the target variable can then be predicted. Understanding how to incorporate patient-specific information into a cross-patient model could potentially benefit the machine learning model as it would receive higher amounts of data and result in higher performance. Some examples of cross-patient models are further discussed in Section 2.2.2.

1.1 Problem Statement and Master Thesis Objective

The approach proposed in this Master's Thesis addresses the challenge of data variability caused by invasive data acquisition procedure of sEEG and differences in neurophysiological responses across patients.

To tackle this, a combination of autoencoders and cross-patient models was developed. A convolutional autoencoder compressed patient-specific data into a unified latent space that captured essential features of the input data. However, ensuring consistency in the feature set alone does not necessarily guarantee semantic alignment between the datasets. To address this, a deep encoder-decoder architecture was implemented, where each patient was assigned an individual encoder to process their unique data, and a shared decoder reconstructed this data into a mel-spectrogram, forming a cross-patient model.

The intuition behind this approach is that the shared decoder, paired with multiple encoders, creates a bottleneck that encourages alignment between different datasets. Additionally, this shared architecture allows the decoder to leverage a larger amount of data, potentially improving its speech reconstruction performance. The objective of the cross-patient model is not to generalize to new, unseen patients, but to leverage the additional data from multiple patients to

enhance predictive power and enforce alignment between different patient datasets.

The decoding model is based on the transformer architecture, which has demonstrated its effectiveness in speech-related tasks such as text-to-speech [14] and speech recognition [15]. Furthermore, the transformer-based models for speech neuroprosthesis are mostly limited to encoder-only variants with the potential of a full encoder-decoder transformer model, being mostly unexplored (see Section 2.2.1 for more).

Implementing an encoder-decoder transformer architecture in this study enables the potential for decoder pre-training. While the encoder will be trained on patient data with a dimensionality specific to each dataset, the target data, represented in a standardized format (e.g., mel-spectrograms), remains consistent across patients. This consistency makes the decoder component interchangeable between models trained on different patients. Therefore, additional experiments will investigate whether pre-training the decoder on data from one patient and subsequently re-using it for other patients can enhance training efficiency and improve predictive performance for new patients.

The research conducted in this Master Thesis contributes to the field of speech neuroprosthesis by investigating the transformer model's capabilities for speech decoding. Additionally, it introduces a cross-patient model designed to enhance speech synthesis across diverse datasets and assesses the potential of pre-trained models for improving model generalizability and performance across patients

1.2 Research Question

These research questions outline the key objectives of this Master's Thesis. The answers are presented in Section 5.5.

- **Is it possible to align different patient datasets in the latent space using a combination of convolutional auto-encoders and a transformer model?**

  Generalizing brain waves across different patients requires that brain signals producing the same or similar outputs (in this research, the output is sound) share common features that describe them. To explore this potential approach for generalization, an autoencoder and a transformer model or a combination of both will be applied to datasets of multiple patients, aiming to align the data across individuals. The quality of this alignment will be evaluated by analyzing the shared latent space of brain signal embeddings from all patients. Since this latent space is high-dimensional, t-SNE will be employed for visualization. An ideal alignment would show a significant overlap between data points originating from different patients, indicating that similar brain signals across different patients are being grouped together. Conversely, no overlap would suggest that the datasets occupy distinct regions in the latent space, implying a lack of alignment.

  Evidence for this research question can be found in Section 4.6, where the results are presented, and in Section 5.3, where those results are discussed.

- **Is an encoder-decoder transformer model capable of efficient speech neuroprosthesis?**

Most current algorithms for predicting speech from brain waves rely on recurrent neural networks. However, transformers have recently been proposed as an alternative, given their state-of-the-art performance in other domains. While encoder- or decoder-only transformers have been explored in research for speech production, the performance of the full encoder-decoder transformer model in this context remains largely unexplored. For this research question, efficient speech production refers to reconstructions that are both intelligible and capable of capturing the unique speech characteristics of each patient. The quality of these reconstructions is evaluated using Mean Squared Error (MSE) and Pearson correlation between the target and reconstructed mel-spectrograms.

Since this research is not primarily focused on optimizing the architecture of the encoder-decoder transformer model, the goal is to demonstrate potential rather than achieve optimal results. Satisfactory outcomes would show the temporal dimension of the reconstruction aligning with the ground truth and the structure of the mel-spectrogram resembling the ground truth. This would correspond to a correlation of approximately 60-70% between the predictions and the test set, with audible inspection confirming that the reconstructed speech retains the recognizable characteristics of the patient's original speech.

The evidence for this question are presented in Sections 4.2 and 4.4 and further discussed in Section 5.2.

- **Is there any improvement gained from training an sEEG-to-spectrogram model on multiple datasets?**

A key objective of this research is to develop models that can utilize data from multiple patients, rather than relying on data from a single individual. Specifically, the study will explore if training with multiple datasets enhances model generalization and performance in synthesizing speech from neural data. Additionally, pre-trained models are relevant to this question, as they are initially trained on data from one patient and subsequently applied to a different patient's data to assess whether prior knowledge transfer improves performance.

To ensure a consistent and fair comparison, the model architecture remains the same across cross-patient, individual, and pre-trained models, with two key differences:

1. The cross-patient model uses multiple encoders—one per patient dataset—whereas individual and pre-trained models have a single encoder.
2. Cross-patient and pre-trained models benefit from access to additional training data, either from multiple patients simultaneously or through pre-training.

The primary metric for this evaluation is model performance on a test set, with each model tested on multiple patients' data for the cross-patient model and on target patient data for the pre-trained model. Using consistent test sets across individual, cross-patient, and pre-trained models allows for a direct comparison. Additional analysis will assess training dynamics, examining trends in training history, overfitting, and underfitting.

This question will be addressed by analyzing the results in Sections 4.2, 4.3 and 4.5, with further discussion in Sections 5.2.2 and 5.2.3.

Organizational structure of the Master Thesis The rest of the work is organized as follows: in Section 2.1, embeddings and the architectures of autoencoders, and transformers are explained. Section 2.2 covers different speech decoding strategies and research in the area of brain embeddings for different BCI tasks. Chapter 3 describes the data, data processing pipeline, model architectures, and latent space visualization. In Chapter 4, the results of all experiments are presented, while Chapter 5 provides an interpretation and discussion of these findings. Finally, Section 5.4 outlines directions for future research, and Section 5.6 concludes this work.

Chapter 2

Preliminaries and Related Work

2.1 Preliminaries

2.1.1 Embeddings

Data is fundamental to any machine learning algorithm, and its representation has a significant influence on the model's performance. While raw data can offer certain advantages, feature engineering is often employed to manually define features that capture key insights from the underlying data[16]. This approach works well for simpler tasks with low-dimensional datasets. However, for more complex problems involving high-dimensional data, allowing the algorithm to automatically learn representations is often more effective than relying on manually crafted features.

Representation learning offers a way to create new data representations that are more informative for machine learning models[16]. These representations, known as latent spaces or embedded spaces, are typically abstract and less interpretable by humans, but they are easier for machines to process and use effectively. Within these latent spaces, embeddings—numerical representations of objects or concepts—capture essential characteristics of the data. The distance between embeddings reflects the similarity between them, which is crucial for understanding relationships in the data. For instance, in Natural Language Processing (NLP), word embeddings capture semantic meanings, such that words like *brain* and *heart* would be close in the latent space, while *brain* and *apple* would be far apart.

Various methods exist for creating latent spaces and embeddings, with Word2Vec being among the most notable[17]. This algorithm produces word embeddings and has significantly advanced NLP. Word2Vec operates using a shallow neural network to learn word associations through one of two classification tasks: predicting a target word given its surrounding context words (the Skip-gram model) or predicting context words given a target word (the Continuous Bag of Words model). Here, context words refer to those frequently appearing near the target word, typically within the same sentence or phrase.

In Word2Vec, each word in the vocabulary is represented by a unique position in an input vector with a length equal to the vocabulary size. For example, with a vocabulary of 1000 words, the input vector will also have a length of 1000, where each position corresponds to a different word. Assuming the hidden layer of the shallow network has 256 neurons, the weight matrix

dimensions will be [1000 × 256]. This weight matrix contains the word embeddings, with each row representing the embedding for a specific word. During training, the network adjusts these weight vectors so that words commonly used in similar contexts become closer together in the latent space, thereby capturing semantic relationships between words. These embeddings, which reflect semantic similarities and relationships, make Word2Vec particularly effective at capturing meaning across various tasks.

Other methods, such as deep metric learning, organize data points in latent spaces based on a distance metric, grouping similar data points while separating dissimilar ones[18]. Studies utilizing this approach are discussed in Section 2.2, where deep metric learning was used to align different datasets within the same latent space.

2.1.2 Autoencoders

Autoencoders are another approach for achieving latent space, as they are designed to find new data representation in an unsupervised manner[19]. Their main objective is to reconstruct the input data, typically through an intermediate layer where new data features are stored. This enables the model to capture essential information for reconstructing the original input.

An autoencoder consists of two main components: an encoder and a decoder. The encoder transforms the data into a compact, intermediate representation, usually by compressing the input into a smaller dimension, while the decoder reconstructs the input data from this compressed feature set. This structure forces the encoder to capture the most critical features of the data within the latent space, where the encoded information resides.

The latent space representation is generally more abstract and less interpretable, but it enables the model to process and utilize data more efficiently. Convolutional autoencoders, in particular, extend this architecture by adding convolutional layers, enhancing the model's ability to capture spatial patterns and further improving its capacity to extract meaningful features.

2.1.3 Transformers and sequence to sequence models

Transformers are sequence-to-sequence models originally designed to solve NLP tasks such as machine translation and text generation[20]. They build on the attention mechanism introduced in earlier Recurrent Neural Networks (RNNs) with encoder-decoder architectures[21], but the transformer places attention at the core of its functionality. A sequence-to-sequence model M_{seq}, in this context, takes a source sequence src as input and is trained to predict a corresponding target sequence tgt.

A default transformer consists of two main blocks: an encoder and a decoder. The encoder extracts essential features from the input sequence, while the decoder uses this encoded information to produce the final output. The decoder receives data from the encoder and also takes as input the target sequence, shifted by one position. This setup enables the use of the teacher-forcing training method, where the model uses the ground truth (rather than its own predictions) to predict the next points in the sequence during training[22]. While this technique allows for parallelization, faster training, and quicker convergence, it can lead to worse performance during inference due to exposure bias[23].

The input to a transformer model consists of a sequence of elements called tokens, where each token represents a discrete point in the sequence. Both *src* and *tgt* are passed through an embedding layer, which converts them into continuous vectors, followed by a positional encoding layer and results in src_{emb} and tgt_{emb}. The positional encoding is crucial because, unlike RNNs, transformers do not inherently capture the sequential order of the data. Therefore, this layer encodes positional information for each token so the model understands where each element is located in the sequence.

Within the encoder block, the data first passes through a self-attention layer. This mechanism allows the model to focus on different parts of the input data. The self-attention mechanism works by creating three matrices: Query (Q), Key (K), and Value (V), all derived from src_{emb}. The dot product of the Q and K matrices produces attention weights, which represent how much focus each token should give to other tokens within the sequence. This QK^T weight matrix contains relevance scores that are processed into an attention mask, determining the contribution of each token when processing the V matrix. The outcome is that the model can focus on the most relevant parts of the input sequence for each prediction.

Additionally, the self-attention layer is divided into multiple heads, where multiple attention mechanisms are applied to different parts of the data. Essentially each attention head processes a different representational subspace of the data, enabling the model to learn multiple aspects or relationships within the data simultaneously. After the self-attention layer, the data goes through an *Add & Normalize* step, where a residual connection combines the original input with the self-attended data, followed by normalization to stabilize training. Finally, the output passes through a feed-forward network with an *Add & Normalize* layer to further refine the information.

The decoder block consists of two stacked multi-head self-attention modules. The first multi-head attention module applies triangular masking, which hides future tokens for each time step, allowing for teacher-forcing and parallelized computation. The second multi-head attention module processes the output of the previous attention layer, with the Q and K matrices derived from the decoder's own output and the V matrix from the encoder. Following each attention module is an *Add & Normalize* layer, which helps stabilize training. The block concludes with a feedforward network, followed by an additional *Add & Normalize* layer.

Both encoder and decoder blocks are repeated N times to form the full transformer model, which ends with a linear and softmax layer applied to the output of the final decoder block.

At the inference stage, the transformer receives the source sequence *src* as input to the encoder and a start token as input to the decoder. The decoder then begins an autoregressive process: it predicts the next token in the target sequence and appends it to the previous tokens. This process continues iteratively until the model outputs a stop token or reaches the maximum target sequence length. During training, the transformer uses teacher forcing to accelerate the training by leveraging the *src* and masked *tgt* sequences, allowing it to generate predictions in a single pass rather than following a multi-step autoregressive approach. Additionally, depending on the application and needs, the transformer model can be used in one of the three configurations: encoder-only, decoder-only, and encoder-decoder.

2.2 Related work

2.2.1 Speech neuroprosthesis decoding strategies

This section reviews some of the approaches for speech neuroprosthesis throughout the years.

Some approaches focus on using Recurrent Neural Networks (RNNs), which are well-suited for processing time-series data.

Anumanchipalli et al.[24] developed a two-stage decoding process to convert ECoG signals into speech. In this approach, two separate bidirectional long short-term memory (bLSTM) RNNs were used: the first model predicted articulatory features from ECoG data, which were then fed into the second bLSTM model to decode these features into acoustic signals. These acoustic features were subsequently synthesized into speech waveforms.

Similarly, Kohler et al.[25] employed a bLSTM to decode spoken sentences from sEEG data inspired by the architecture of a text-to-speech model Tacotron-2[26]. Their RNN architecture was equipped with an attention module and an encoder-decoder framework, further enhanced by pre-nets and post-nets to refine predictions. The input data was initially processed by a convolutional neural network (CNN), which was then fed into the RNN, with the output being mel-spectrogram coefficients. These coefficients were synthesized into speech using a WaveGlow network. The reported results scored from around 71% to around 81% Pearson correlation depending on the dataset used.

In contrast, Angrick et al.[27] utilized a modified DenseNet model to predict mel-spectrograms from neural data, which were then synthesized into audio waveforms using a WaveNet vocoder. Their model used 3D convolutions to account for the temporal dimension and how electrode signals evolve over time. This approach was applied to an ECoG dataset of single words and achieved from around 20% Pearson correlation to around 70% depending on the dataset used.

Other approaches explore the possibility of using transformer-based models in speech decoding tasks. Shigemi et al.[28] hypothesized that a transformer model could outperform a bLSTM in speech reconstruction tasks. In their work, ECoG data from participants reading sentences was first processed by a CNN with temporal filters, and the output was fed into a transformer model trained to predict mel-spectrograms. These predictions were then synthesized into audio using a parallel WaveGAN, and the transformer model outperformed the bLSTM model in their experiments achieving around 68% Pearson correlation.

Chen et al.[29] applied a transformer variant called the Shifted Windows (SWin) transformer, comparing its performance to that of ResNet and bLSTM models. Unlike traditional transformers that process entire input sequences simultaneously, the SWin transformer divides the input into separate patches[30]. Attention is calculated within each patch individually, followed by a merging process that combines neighboring patches, where attention is recalculated. This method allows the model to capture local patterns while integrating global information. The data for this study was recorded using ECoG, and speech reconstruction involved patients performing reading tasks, where they read the same words multiple times. Their speech production pipeline was divided into two main parts. In the first part, a speech encoder and synthesizer were trained to extract speech parameters from a spectrogram and then reconstruct the spectrogram. Each patient had a pair of speech encoders and synthesizers. In the second part, their transformer decoder predicted various speech parameters, which were used by a speech synthesizer to create

mel-spectrograms. Finally, these spectrograms were used to reconstruct speech into an audio waveform. The results showed that the ResNet achieved the highest performance in speech reconstruction, closely followed by the proposed transformer decoder.

Finally, Wu et al. [31] applied a full encoder-decoder transformer model to reconstruct single words from sEEG data and compared its performance to a linear regression model and an RNN. The model architecture was based on the vanilla model from the original transformer paper [21] with a modification to accommodate for speech synthesis setting. Their findings suggested that the reconstructed speech was not intelligible.

As mentioned the goal of speech neuroprosthesis is to create decoding models and pipelines capable of real-time speech synthesis. The aforementioned studies were mostly concerned with the feasibility and possible improvements in speech decoding strategies. Building a real-time speech decoding model is a more complex problem; however, in a more recent study titled "An Accurate and Rapid Neuroprosthesis,"[32], the authors aimed to build a decoding brain-to-text model that can work in real-time. Their decoding pipeline used an RNN to predict phonemes every $80ms$. This sequence was then fed into a language model to predict the most likely word, and a sequence of words was fed into a different language model tasked with predicting the most likely sentence. The decoding accuracy ranged from 99% to 90% depending on the vocabulary size and training time.

Producing models for people who lost their ability to speak is also an important challenge that was addressed by Metzger et al. as they attempted to develop a speech decoding model for a patient suffering from severe paralysis and anarthria who had lost the ability to speak. Since the patient could not provide speech data, the researchers employed a pre-trained Hidden-Unit BERT (HuBERT) model to generate a reference training set. To achieve this, a healthy participant read the same sentences as the target patient, and the obtained speech data was transformed into speech units using HuBERT.

Meanwhile, the target patient participated in an attempted speech task in which they tried to read the same sentences as the healthy participant, allowing researchers to capture the patient's neural activity during these attempts. Using this combined dataset — speech units from the healthy participant and corresponding neural data from the patient — the researchers trained an RNN model to predict speech units from the patient's neural signals in a multi-class classification task.

Once the RNN model successfully predicted the sequence of speech units, another machine learning model was used to convert these units into a mel-spectrogram, which was then transformed into a speech waveform. To assess model performance, human listeners transcribed the synthesized speech, and the resulting transcriptions were compared to the ground truth. The Word Error Rate (WER) was used as the evaluation metric, with the decoding model achieving a WER ranging from 8.20% to around 60% depending on the dataset used.

2.2.2 Neural embeddings and cross-patient research

This section reviews studies on neural data embeddings and cross-patient model development, highlighting alternative approaches to generalizing brain wave patterns across different individuals. Three distinct BCI applications are examined: motor classification, emotion classification, and speech neuroprosthesis. The first two applications focus on classifying neural data into

predefined categories within motor or emotional domains. In contrast, speech neuroprosthesis involves a regression task, which could be simplified as a classification problem.

Motor classification Guetschel et al. proposed a method for creating shareable embeddings between patients for a motor classification task[33]. They utilized a modified EEGNet[34], excluding its final softmax layer, and trained it with a triplet loss function to produce embedding vectors from the input data. Once the embedding model was trained, these embeddings were used in a classification task involving a simple logistic regression model. The data for this study came from a High-Gamma Dataset, which maintains a consistent channel count across multiple subjects. For evaluation, they employed a Leave-One-Subject-Out (LOSO) procedure, where all subjects except one were used to train the embedding function and the classifier, with the remaining subject used to test the classifier. They also tested a variation of this approach, where a portion of the left-out subject's data was used to train the embedding classifier. Their results demonstrated that in the complete LOSO scenario, overall performance decreased. However, in the partial LOSO scenario, the performance improved beyond the baseline, indicating that generating embeddings with triplet loss can enhance prediction accuracy.

Autthasan et al.[35] developed a multi-task autoencoder model designed for motor imagery classification in both subject-dependent and subject-independent settings. The autoencoder was trained using three loss functions: the standard reconstruction loss (mean squared error), the triplet loss, and the cross-entropy loss. The triplet loss was employed to preserve distinguishable patterns in the latent space, while the cross-entropy loss optimized the latent space for classification. The classification was performed using a fully connected layer followed by a softmax activation. In the multi-subject setting, the model was trained using LOSO cross-validation, where all subjects except one were used for training, and the remaining subject was used for testing. Their results demonstrated that the performance of the subject-independent model was comparable to baseline models (including EEGNet) on two datasets, and it outperformed the baseline models on one dataset.

Emotion classifcation In their study, Arjun et al.[36] developed an LSTM-based autoencoder with channel attention to create a latent space representation of EEG data, which was then used as input to a cross-patient CNN classification model with attention. The input EEG data had a consistent channel count across all patients, allowing for straightforward integration into the model. The researchers evaluated their pipeline using a LOSO testing procedure on a binary classification task. Their results demonstrated the highest classification performance, though they did not provide a comparison with models trained on individual patients.

Shen et al.[37], on the other hand, proposed an inter-patient pipeline designed to align datasets from multiple patients. Their approach involved an encoder to extract meaningful features and map the data into a latent space, followed by a projector that refines this latent space and uses contrastive learning to create alignment between different patients. For classification, the encoder generates latent representations that are subsequently fed into a classifier to predict emotion labels. Their pipeline performed well on unseen patients and emotions during LOSO testing, demonstrating strong generalization across both new subjects.

Speech neuroprosthesis In[38] the authors developed a cross-patient model for speech production using ECoG and sEEG data. Despite variations in the number and placement of recording electrodes across patients, the model was designed to leverage this spatial information, en-

abling it to process data from an arbitrary number of channels and patients.

Their decoding pipeline was an extension of their previous work[29] and it was split into two parts. Therefore, as in the previous study, the first part contained a speech encoder and synthesizer for each patient, where the models were tasked with reconstructing spectrograms from extracted speech parameters. The second part of the pipeline employed a variant of the SWin transformer. To create patches, the input data was segmented along the temporal dimension, with each token representing a signal from a single channel. Attention was calculated across all available channels and their neighboring tokens in the time dimension. The spatial information from the electrodes was incorporated into the model via a positional bias. This bias was determined by transforming the difference between the standardized coordinates of two electrodes and the temporal difference in their signals using a multi-layer perceptron (MLP). The neural decoder then predicted speech parameters from the neural data, which were fed into the trained speech synthesizer to produce a spectrogram, ultimately reconstructing the speech.

The cross-patient model performed worse on unseen subjects than subject-specific models; however, their results indicate the feasibility of creating cross-patient models using the proposed method and on limited data.

While not directly focused on neural data embeddings, the work of Metzger et al. [9], described in the 2.2.1 Section, has shown HuBERT's ability to generalize speech across patients.

Chapter 3

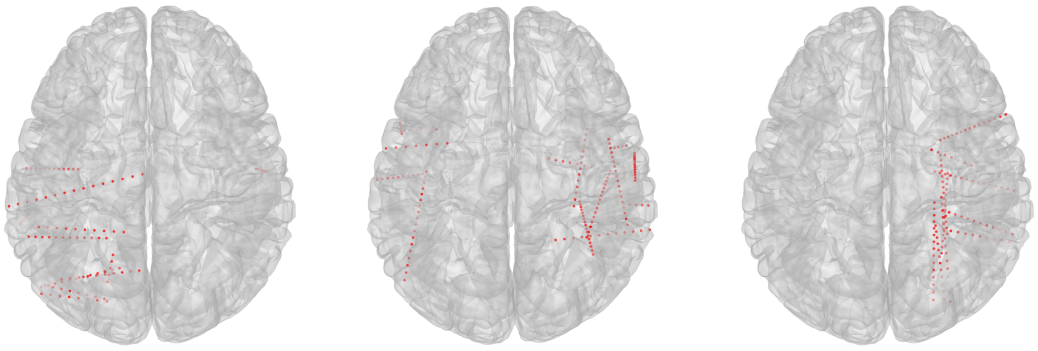
Data and Methodology

The code for the preprocessing pipelines, decoding models and results can be found at the following GitHub Repository.

3.1 Data

The data for this research was collected from 10 patients who had intracranial electrodes implanted in their brains in order to locate epileptic zones. During data collection, those patients participated in a sentence reading experiment where they read 100 sentences from the Mozilla Common Voice Dutch corpus[39], with each sentence followed by a 2-second resting period. While each patient read the sentences, the neural signals were recorded at 1024 Hz while the audio was captured at 48000 Hz.

The number and location of the electrodes varied among patients due to the differing locations of epileptic zones in the brain. Additionally, each electrode had multiple recording points, referred to as channels, as illustrated in 3.1. Essentially, each channel records neural activity from a different brain area.



(a) Patient 00 (b) Patient 01 (c) Patient 08

Figure 3.1: **Electrode placement for different patients.** Each patient has a varying number of electrodes that are placed in different locations. Each electrode has multiple recording points (i.e. channels), as shown by the red dots.

3.2 Data pre-processing

The recorded neural signals were first synchronized with the corresponding audio data and any channels that captured signals unrelated to brain activity, such as EKG^+ or $MKR2^+$ were removed.

Neural processing While various frequency bands (e.g., alpha, beta, delta) are commonly used in machine learning applications for computational neuroscience and BCI [40], high gamma features (HGF) have shown a strong correlation with cognitive functions, particularly speech production[4]. Due to this, they have become a key component in the development of speech neuroprostheses. HGF are typically characterized by broadband gamma activity within the $70 - 170$ Hz range.

To isolate the high gamma band from the raw neural data, a band-pass filter targeting frequencies between 70 and 170 Hz was applied. In order to minimize the effects of line noise and attenuate the first and second harmonics, two additional band-stop filters were designed, covering the frequency ranges of $98 - 102$ Hz and $148 - 152$ Hz, respectively. After filtering the signal, it was segmented into overlapping windows of approximately 50 ms, with a frameshift of about 1 ms. Within each window and channel, to capture the neural activity, the logarithmic power was computed using the following equation:

$$log(\sum(W_c))^2 + 0.01 \tag{3.1}$$

where W is the activity within a specific time window, while c denotes the corresponding neural channel. Finally, the signal was normalized across the temporal dimension using z-score normalization to ensure a standardized scale across all features.

Audio processing For audio processing, a pre-trained neural vocoder based on the VocGan architecture [41] was used to extract a logarithmic mel-spectrogram representation of the audio waveform. This feature is generally used in machine learning for audio applications[42] and as noted by Silva et al.[4], this speech format is more similar to how human auditory systems work making it a common audio feature in speech neuroprosthesis. Prior to using the VocGan, the raw audio was resampled to match the expected input sampling rate of the VocGan model. Specifically, the audio was resampled to a lower frequency of $22,050$ Hz. Once resampled, the audio was scaled to a 16-bit integer range for processing. The 80-bin log mel spectrogram was then extracted using the pre-trained vocoder, configured with the same window size ($\sim$ 50 ms) and frameshift ($\sim$ 1 ms) as those used for neural processing. Additionally, the neural vocoder was employed in the final stage of the speech decoding pipeline to synthesize the audio back into waveform format from the extracted mel-spectrograms.

This pre-processing pipeline resulted in two signals: source sequence of sEEG denoted as *src* and target sequence of mel-spectrogram denoted as *tgt*. Both signals were aligned in the temporal dimension and had the same sequence length. Their feature sets describe either the set of possible recording channels (from the recording electrodes) for the sEEG signal or the mel-spectrogram bins for the mel-spectrogram. The dimensionality of the former varies between patients (from 44 channels to 127 channels), while the dimensionality of the latter stays constant at 80 bins for all patients. Finally, to prepare the data for machine learning models, both the *src* and *tgt* sequences were further divided into I overlapping windows, distinct from the windows used to compute neural or audio features in the previous processing step. Each window contained 100

time points, with an overlap of 5 time points between consecutive windows. Figure 3.2 illustrates this pre-processing pipeline.

Audio classification For visualising and evaluation purposes it was crucial to determine the amount of speech signal present within a mel-spectrogram window. First, for the entire mel-spectrogram sequence (prior to the second windowing), the energy across all frequency bins was averaged for each frame:

$$\text{melSpecAvg}_t = \frac{1}{B} \sum_{b=1}^{B} \text{melSpec}_{t,b} \tag{3.2}$$

where B represents the number of mel-spectrogram bins, and $melSpec_{t,b}$ denotes the mel-spectrogram value at time t and bin b. Additionally, all frames of $melSpecAvg_t$ were averaged to obtain μ:

$$\mu = \frac{1}{T} \sum_{t=1}^{T} \text{melSpecAvg}_t \tag{3.3}$$

wher T denotes the length of the entire mel-spectrogram sequence. Next, each frame t was classified as speech or silence based on whether the average energy $melSpecAvg_t$ exceeded the μ value:

$$\text{label}_t = \begin{cases} 1 & \text{if melSpecAvg}_t > \mu \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

After applying this labelling process, each time-point within a mel-spectrogram window was assigned its corresponding label. Calculating the speech percentage in a specific mel-spectrogram window was done by dividing the number of speech labels by the total length of the window. Examples of this labelling process can be found in Appendix A.1.

3.3 Models architectures

3.3.1 Convolutional Autoencoder

To recreate the sEEG signal and compress the information into a latent space, a convolutional autoencoder architecture was employed. The encoder consisted of five convolutional layers with an increasing number of filters $(5, 10, 15, 30, 30)$ and a final linear layer. An increasing number of filters allowed to capture different aspects of the signal.

The first three convolutional layers used progressively larger kernel sizes $(3, 5, 10)$, performing temporal convolutions across each channel independently to capture channel-specific information. In the fourth convolutional layer, both spatial and temporal convolutions were applied with a kernel size of 15 for the temporal dimension and E_p (the number of electrodes for patient p) for the spatial dimension. This combined all electrode signals into a single sequence, reducing the spatial dimension to 1. Since the electrode placements were random with no inherent spatial order, no specific spatial arrangement was assumed in the data. Although methods such as cyclic convolutions have been proposed to address this issue [43], they were not necessary for this particular application.

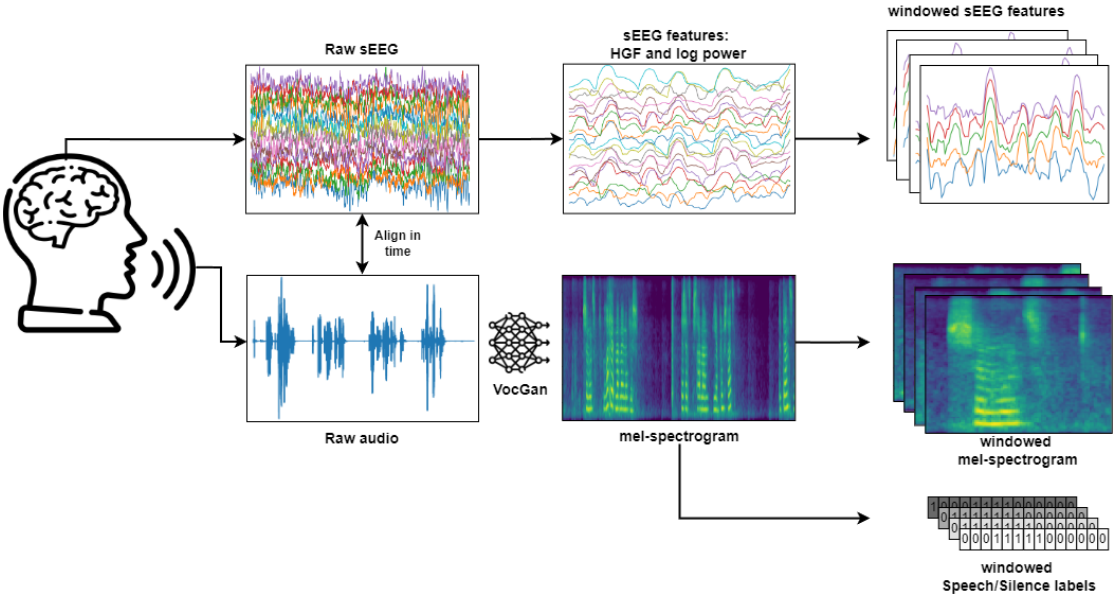


Figure 3.2: **Pre-processing pipeline for neural and audio data.** During data collection, patients read sentences while both neural (sEEG) and audio signals are simultaneously recorded. The first step is the temporal alignment of the two modalities to ensure synchronization. In the neural data pipeline, high-gamma features (HGF) are extracted from the raw sEEG signals, which are then summarized using overlapping windows in which the log power is calculated. In the audio data pipeline, the raw audio is processed through a pre-trained VocGAN model, converting it into a mel-spectrogram. Both the processed neural and audio features are further divided into equal-length, overlapping windows. Additionally, regions of speech and silence in the mel-spectrogram are labeled and segmented into windows that correspond to the windowed mel-spectrograms.

The fifth and final convolutional layer applied a temporal convolution with a kernel size of 21. Each convolutional layer used a stride of 1 and no padding. The increasing kernel size was implemented to initially capture local patterns in the data, allowing the model to detect finer details or short-term dependencies. As the kernel size increases, these smaller patterns are progressively combined into larger and longer patterns, enabling the model to capture broader, more complex temporal dependencies or structures within the data. After each layer, a ReLU activation function was applied, followed by batch normalization to stabilize and improve the training process. The final linear layer projected the 30 filters into a 64-feature set, resulting in a $[50 \times 64]$ sequence, where the first dimension represents time and the second dimension corresponds to the latent space features.

The decoder mirrored the architecture of the encoder with the convolutional layers switched to transposed convolution layers. This autoencoder architecture is visualized in Figure 3.3. To train the model, the standard Mean Squared Error (MSE) loss function was used, along with the Adam optimizer, which was set with an initial learning rate of 1×10^{-2}. Additionally, to enhance convergence, the learning rate was reduced once the model's performance plateaued.

Early stopping was also applied with patience[1] of 10 epochs, ensuring the training process would halt once the model had fully converged, preventing overfitting or unnecessary training cycles. The reconstructions of sEEG data and general performance of the autoencoders can be found in section 4.1.

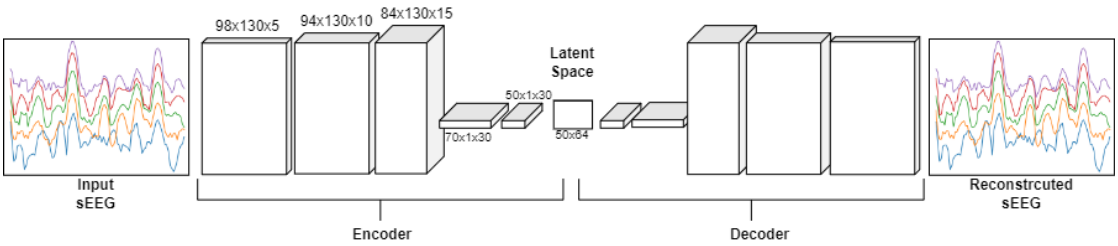


Figure 3.3: **Convolutional Autoencoder Architecture.** In the encoder, each layer applies an increasing number of filters, with the first three convolutional layers focusing on capturing local temporal patterns within each channel using progressively larger kernel sizes. The fourth convolutional layer performs a joint convolution across both time and channels, collapsing the channel dimension. Following this, an additional temporal convolution is applied, and a linear layer projects the extracted features into the latent space. The decoder reverses this process, using transposed convolutions to reconstruct the original sEEG data from the latent space representation. The numbers denote the dimensions of the data.

3.3.2 Neuroprosthesis models

The following models took a sequence of high-dimensional sEEG data as input and were tasked with predicting a sequence mel-spectrograms of the same length as the input data. For the transformer-based models, there were three variants:

- **Individual Encoder-Decoder Transformer**: This variant was used for training on individual patients.
- **Cross-patient Encoder-Decoder Transformer**: This variant was employed for training multiple patients on a semi-single model.
- **Pre-trained Encoder-Decoder Transformer**: This variant was used for training individual as well as cross-patient models utilizing pre-trained decoders.

Encoder-Decoder Transformer architecture The transformer model used in this research is based on the original encoder-decoder transformer architecture [20], which was adjusted to the speech neuroprosthesis setting. Both the inputs to the encoder and decoder are passed through a linear layer which serves as an embedding layer, after which standard positional encodings in the form of a sinusoidal function are added to src and tgt. These encodings are calculated using the following formulas:

$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{3.5}$$

[1] *Patience* in early stopping refers to the number of epochs the model will wait for an improvement in the monitored performance metric before stopping the training process.

$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{3.6}$$

where, i refers to the token dimensions, pos refers to the token position and d_{model} is the dimension of the embedding.

The encoder consists of three layers, each composed of multi-head attention and feedforward network blocks, followed by add and normalization layers. Similarly, the decoder also has three layers, but its blocks include masked multi-head attention, encoder-decoder multi-head attention, and a feed-forward network, all followed by add and normalization layers. Finally, the output of encoder is passed through a linear layer, which converts the decoder output sequence of dimensions $[100 \times 256]$ to the target sequence of dimensions $[100 \times 80]$. To prevent over-fitting, the model used dropout layers after each attention and feed-forward layer, as well as in the embedding and position encoding layers. The model architecture can be also found in Figure 3.4. To summarize, the parameters of the transformer architecture were as follows:

- **encoder layers:** 3
- **decoder layers:** 3
- **embedding size:** 256
- **feed-forward network dimension:** 256
- **number of attention heads:** 8
- **dropout:** 0.5

Cross-patient Transformer This variation of the encoder-decoder transformer model was designed to align the latent space across different patients and use more training data for speech synthesizes. The model comprises multiple encoders (one for each patient) and a shared decoder. During training, the model iterates through different datasets and encoders in each training batch while keeping the decoder shared. Figure 3.5 illustrates this model's architecture. The architecture of this network besides multiple encoders, was the same as the individual model.

The training loop is structured to accommodate the largest dataset by iterating over the smaller datasets multiple times if necessary. For instance, consider a shared model between patient A, with 100 training batches, and patient B, with 50 training batches. The model would have two encoders and one decoder. The training loop would run for 100 iterations, processing patient A's dataset once and patient B's dataset twice. In each iteration, both encoders and the shared decoder are exposed to batches from both patients' datasets.

Pre-trained Transformer The pre-training strategy involved initially training an individual transformer model on a specific patient dataset. Once trained, the decoder from this model was then reused in a different individual or cross-patient model. In the target model, the reused decoder was paired with encoder components trained on data from different patients than those used in the initial pre-training stage. This model is shown in Figure 3.6 This approach aimed to test the possibility of transferring knowledge and improving the predictions for new patients datasets.

Model training and evaluation During experimentation, it was observed that the transformer models were not learning effectively, resulting in unusable predictions (further discussed in section 5.2.4). To mitigate this issue, teacher forcing was disabled, altering the training process to resemble the evaluation phase more closely. In this modified approach, the encoder first processes the entire sEEG sequence. The decoder then starts decoding with a start token and the encoded sequence. The start token was represented as an array of dimensions $[1 \times 80]$ filled with zeros which was done to mimic silence before speech. The decoder continues predicting the subsequent frames in an autoregressive manner until it reaches the target sequence length.

The input data to any of those transformer models was either the preprocessed HGF sEEG data or the same HGF data but encoded by a fully trained autoencoder (see Figure 3.7). The latter configuration will be referred to with a prefix of latent, e.g., latent transformer. The data was split into training, testing, and validation sets with a 70%, 15%, and 15% split, respectively. The cross-patient model was evaluated on each patient separately with the same testing split. No shuffling was applied, as the data was organized into overlapping windows and shuffling would have caused data leakage from the training set into the testing and validation sets. Although cross-validation could have provided more accurate results, without teacher forcing, training the transformer model became significantly slower, making even a 5-fold cross-validation impractical. While the speech synthesis models were trained to minimize MSE loss, Pearson correlation was used to evaluate the models' performance as this metric is commonly used to assess the quality of mel-spectrograms in speech neuroprosthesis research [4]. To compute the correlation between a predicted mel-spectrogram and its ground truth, the correlation between all spectral bins was first calculated and then averaged. The Pearson correlation for a specific spectral bin was calculated by the following equation:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.7}$$

Here, $\bar{x}$ and $\bar{y}$ are the means over all time points of a specific spectral bin for the prediction and ground truth, while x and y represent individual time points within the same spectral bin.

Additionally, in the later stages of this research, it was discovered that Pearson correlation does not perfectly reflect models predictive capabilities. Since this discovery came late in the project, a thorough investigation into alternative metrics was not feasible. However, the Mel Cepstral Distortion (MCD) metric [44] was introduced, as it provides a more relevant measure for capturing the perceptual quality of speech by quantifying the difference in mel-spectrogram features [4].

Therefore, for most of the experiments, the MSE and Pearson correlation are reported, while MCD is mainly used for results presented in Section 4.4.

All individual transformer models were trained for 500 epochs, while the cross-patient variants were trained for 200 epochs. This was usually enough to achieve convergence over validation data.

Each model's performance and test set results can be found in section 4.2. Models found in this overview were trained for varying numbers of epochs: 500 epochs for individual models, 200 epochs for cross-patient models, and 1000 epochs for pre-trained models. In the case of the pre-trained models, 500 epochs were used to train an individual model on the patient 01 data. Afterwards, the decoder from this trained model was extracted and incorporated into the

pre-trained model, which was then trained for an additional 500 epochs.

Beyond overall results, different aspects of each model are presented. The training history, predictions, and evaluation outcomes of the individual model trained on patient 01 data are provided in Section 4.4, as this model achieved the best performance across all models. Additionally, this model was trained for 5000 epochs instead of 500 epochs. For the cross-patient models, only the training history is shown in Section 4.3. As for the pre-trained model, both its training history and predictions are included in Section 4.5. In this instance, an individual model was first trained on the patient 01, and upon completion, its decoder was used in the pre-trained model, which was further trained on the patient 06 data.

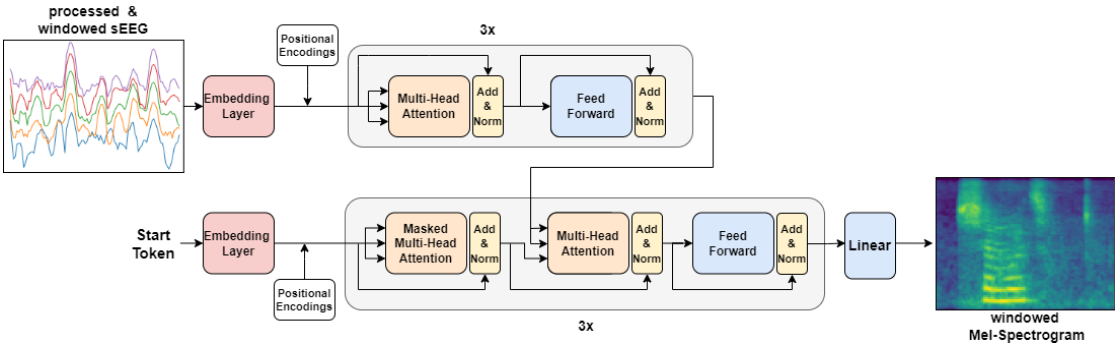


Figure 3.4: **Individual Network Architecture.** This encoder-decoder transformer is designed for training on data from a single patient. Both the encoder and decoder first process their respective inputs through an embedding layer, followed by the addition of positional encodings. The encoder is composed of 3 layers, each featuring multi-head attention modules and feedforward networks, which capture and encode the neural signal. The decoder, similarly structured with 3 layers, utilizes masked multi-head attention, encoder-decoder attention, and feedforward modules. A start token (an empty array) serves as the initial input to the decoder, representing the absence of speech prior to its generation. The decoder's output is passed through a linear layer, transforming it into mel-spectrogram feature space. While not illustrated in the figure, the decoder operates in an autoregressive manner: after predicting each frame of the mel-spectrogram, that frame is fed back as input to the decoder for the next prediction, continuing until the entire sequence is generated. Architecture inspired by Vaswani et al.[21].

3.3.3 Linear Regression

During the autoencoder training, a linear regression model M_{LR} was used to assess how well *tgt* could be reconstructed from either the latent representation of *src* or the reconstructed *src*. As the autoencoder compressed the input sequence of neural data, M_{LR} provided a straightforward method to measure the amount of information lost during the encoding and decoding process. Ideally, the information loss across input data, latent data, and reconstructed data would be minimal. Additionally, a separate M_{LR} was used to establish a baseline for speech synthesis from sEEG data.

In both cases, the M_{LR} was trained and evaluated using 70%, 15%, and 15% split for training, validation, and testing data, respectively, to match the training and evaluation procedure for the

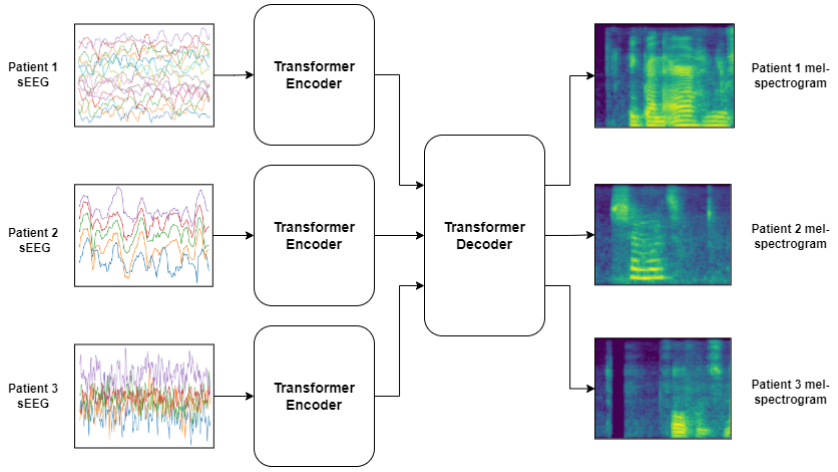


Figure 3.5: **Cross-patient network architecture.** The model consists of multiple encoders, each trained on sEEG data from different patients, all connected to a shared decoder. The decoder, in turn, is trained on mel-spectrogram datasets across these patients. During training, the encoders alternate utilizing the decoder, enabling the decoder to generalize across multiple patient datasets simultaneously.

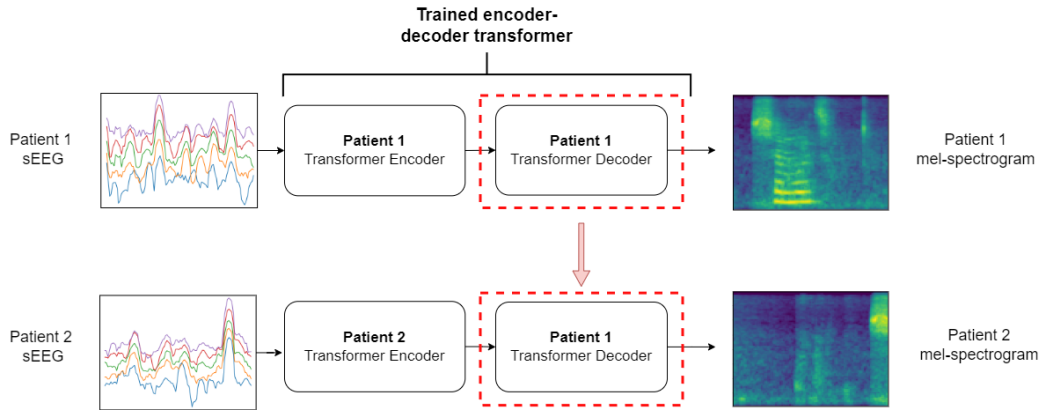


Figure 3.6: **Pre-trained network architecture.** The model employs a decoder pre-trained on data from a different patient. In the top section, the individual model is trained on Patient 1's sEEG data, where both the encoder and decoder are trained on Patient 1. In the bottom section, the pre-trained model employs the decoder from the Patient 1 model, with the encoder built for Patient 2's sEEG data.

transformer models and ensure more accurate comparisons. The results for both M_{LR} can be found either in 4.1 or 4.5.

3.3.4 Visualzing the embedding space

The latent space alignment between different patient datasets was evaluated using t-distributed stochastic neighbor embedding (t-SNE). T-SNE is a dimensionality reduction technique that is commonly used for visualizing high dimensional and non-linear data, as it preserves the local

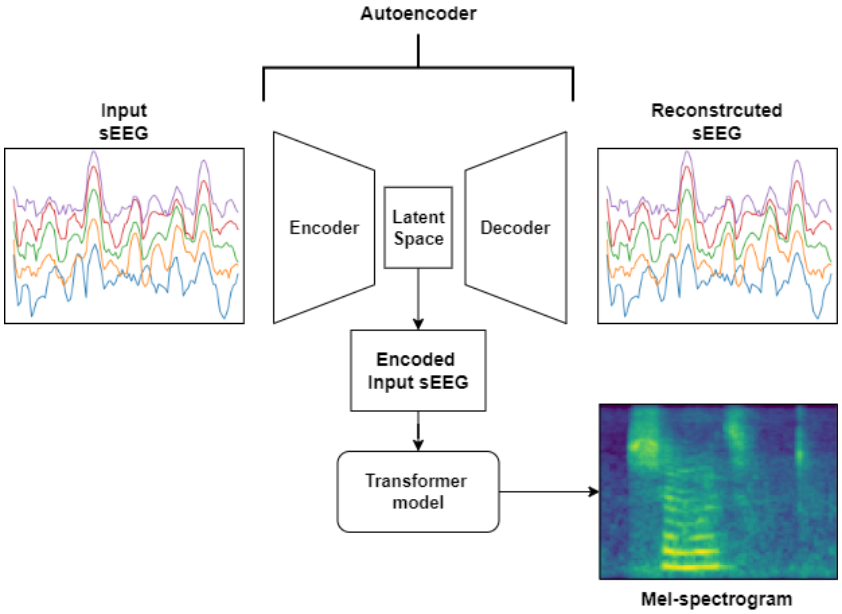


Figure 3.7: **Latent Transformer-Based Model**. The input neural data (sEEG) is first used to train an autoencoder, which learns to encode and reconstruct the original input data. Once the autoencoder is trained, it encodes the entire dataset into the latent space representation. This encoded neural data is then used as input to train an transformer-based model, which could be one of three models: the individual model, the cross-patient model, or the pre-trained model.

and global structure of the data[45].

Latent space extraction To perform this visual analysis, the latent spaces for each patient had to be first collected. As outlined in the first research question (see 1.2), the encoded data was extracted using one of three possible model configurations:

1. autoencoder
2. cross-patient transformer
3. autoencoder + cross-patient transformer (latent cross-patient transformer)

In the first and second cases, the latent space was extracted by encoding the data using the encoder component of the respective models. In the third case, the latent space was extracted from the cross-patient transformer encoder, which had been trained on data previously encoded by a fully trained autoencoder (see Figure 3.7). Additionally, in both cases where an autoencoder was used, each patient had their own individual autoencoder, which was trained until convergence. The cross-patient transformers were trained for 200 epochs.

t-SNE Latent Space Visualization After extracting latent spaces for each patient, the data was concatenated into an array N, and the t-SNE algorithm was applied to this grouped data. t-SNE was used to reduce the dimensionality of the latent space to 2D for two cases:

- Trajectory Lines: with the temporal dimension intact
- Points: with the temporal dimension averaged

Trajectory Lines To calculate the trajectory lines, the first and second dimensions of the array $[N \times 100 \times 256]$ were collapsed, resulting in an array of dimension $[(N * 100) \times 256]$. This array was normalized using a z-score normalization before applying t-SNE. The output provided a 2D representation of the trajectories of brain signal sequences over time for each patient's latent space.

Points For the second case, where the temporal dimension was averaged, the second dimension of the combined data was averaged, producing an array of dimension $[N \times 256]$. Similar to the trajectory lines, this array was normalized and then subjected to t-SNE, providing a 2D plot of averaged points from the latent space of each patient.

The results for the visual analysis of the latent space can be found in section 4.6.

Chapter 4

Results

4.1 Autoencoder results

Figure 4.1 shows the MSE results for the autoencoders trained on different patients. The performance of these autoencoders varies based on the patient with most autoencoders achieving MSE values between 0.7 and 0.8. Autoencoders trained on patients 00 and 12 performed best, at around 0.41 and 0.44 MSE, respectively, while autoencoder trained on patient 10 performed the worst, with an MSE of approximately 0.98.

Figures 4.2 and 4.3 demonstrate the reconstruction power of autoencoders for patients 00 and 01. For patient 00, the reconstructions closely match the input data with the model struggling only with sharp dips and peaks as evidenced by channels LO3 or LT1.

In contrast, the reconstructions for patient 01 are less accurate. The autoencoder model struggles to reconstruct or even generalize the signal in most cases. While it follows the signal closely in certain time intervals for channels like LO2 and LO3, overall, it either provides poor generalization (as seen in channels RO4, RY9, or RY11) or fails to capture the signal characteristics entirely (channels LT13 or RM6).

Figure 4.4 compares the performance of a linear regression model trained on one of three types of data: the original data, the encoded data (latent space), or the reconstructed data. For patient 00, the baseline linear regression performance starts at a correlation of around 20%. As training progresses, the performance over latent and reconstructed data improves, with the performance on reconstructed data matching the baseline, while the performance on latent data lags behind the baseline by about 10%. For patient 01, the pattern is slightly different, as the baseline performance starts at around 28% and the performance over latent data quickly matches it, while the performance of the reconstructed data surpasses the baseline almost immediately and achieves a correlation of around 38%.

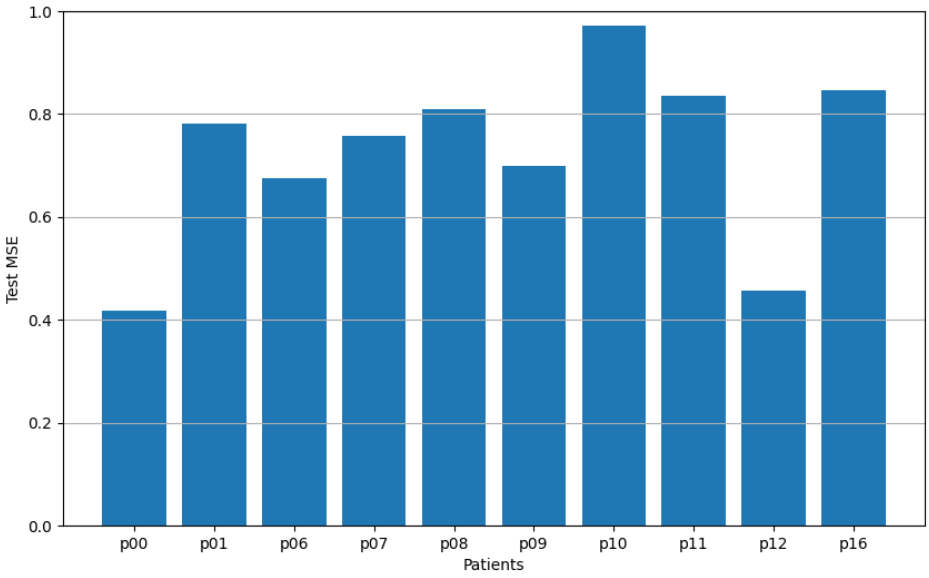


Figure 4.1: Autoencoder MSE results over the test set of different patient datasets.

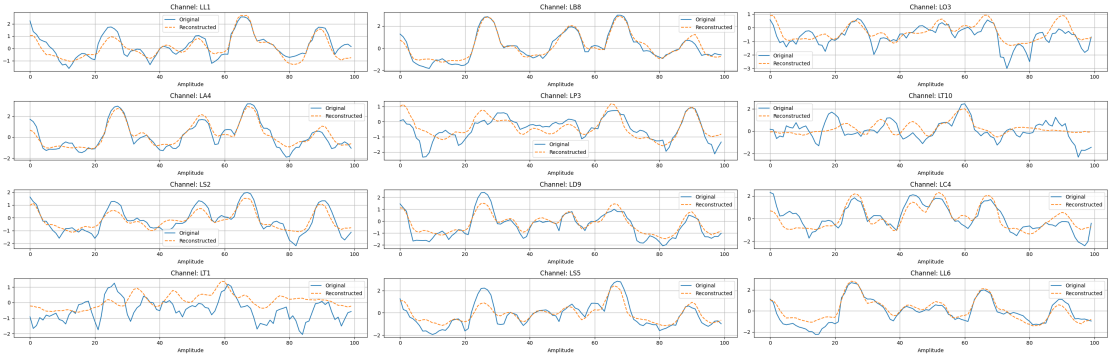


Figure 4.2: sEEG input data reconstructed by an autoencoder trained on patient 00 data. Each subplot in this figure represents a signal for a different channel, with the blue line representing the input data and the orange line representing the reconstructed signal.

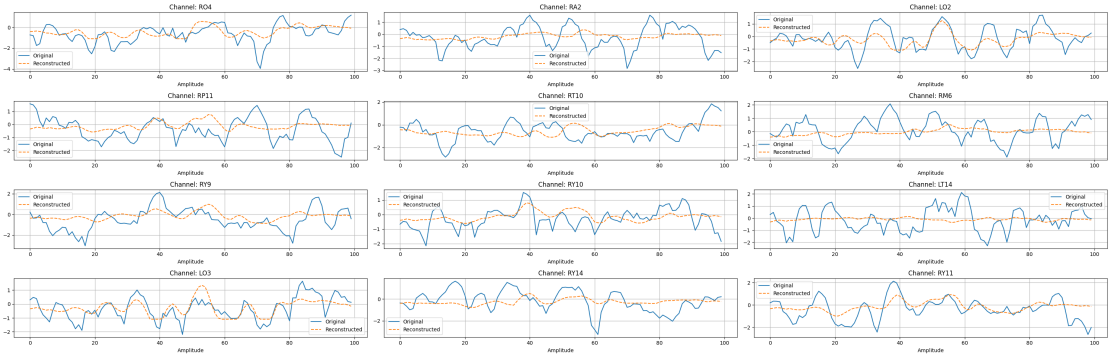
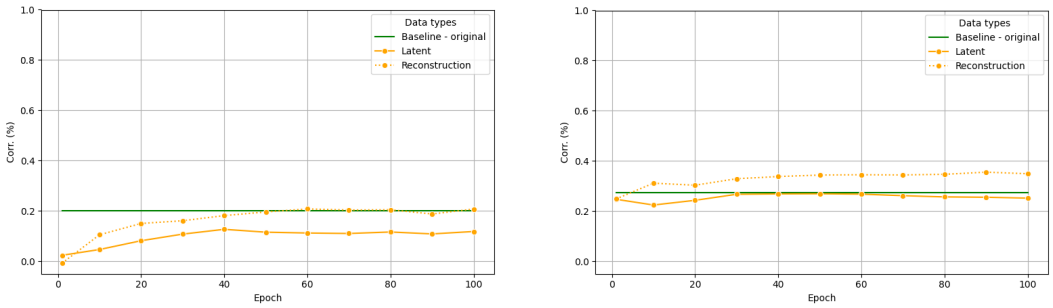


Figure 4.3: sEEG input data reconstructed by an autoencoder trained on patient 01 data. Each subplot in this figure represents a signal for a different channel, with the blue line representing the input data and the orange line representing the reconstructed signal.



(a) p00 (b) p01

Figure 4.4: Speech neuroprosthesis results of a linear regression (LR) model trained on data encoded or reconstructed by an autoencoder for different patients at various stages of autoencoder training. The three lines represent the performance of the LR model trained on different input data: the green line shows the performance when trained on the original data, the solid orange line represents the performance when trained on the latent data encoded by the autoencoder, and the dashed orange line shows the performance when trained on the data reconstructed by the autoencoder. The x-axis tracks the autoencoder's training progress in epochs, while the y-axis measures the correlation between the predicted and ground truth mel-spectrograms on the test set. (a) Autoencoder and LR trained on patient 00. (b) Autoencoder and LR trained on patient 01.

Figure 4.5: Comparison of model performance across different datasets using various models: **LR**, **latent LR**, **transformer**, **latent transformer**, **cross-patient**, **latent cross-patient**, and **latent transformer**. The models were evaluated on a test set; top plot shows the correlation results, while the bottom plot shows the MSE results.

4.2 Transformer overall results

As shown in Figures 4.5 and 4.6, the LR model performed the worst out of all of the tested models. The individual models performed the best for each patient, closely matched by the cross-patient models or the pre-trained models. There are some exceptions to this; for example, for patient 09, the latent transformer outperformed other models, and for patient 12, all models performed relatively the same. One common trend that can be observed is that generally, models trained on the latent data performed worse than models trained on the normal data.

Overall, the highest correlation was achieved over the patient 01 testing set, as its individual model as well as cross-patient model settled at around $47\% - 48\%$. This is closely followed by the patient 06 data, for which the individual, cross-patient, and pre-trained model achieved a correlation of around $42\% - 44\%$. The worst performance was achieved over patient 08 and 10 data, where the correlation ranged from 2% to 17% depending on the used model.

4.3 Cross-patient model - training history

Figures 4.7 and 4.8 present the training histories for the normal and latent cross-patient models. In the case of the normal cross-patient model, the validation correlation rapidly stabilizes around 30%. Furthermore, for some patients, such as 00, 01, and 06, the validation correlation increases over time, indicating gradual improvement. Conversely, for patients like 07 and 11, the validation correlation declines as training progresses, signalling potential issues in generalization. The MSE loss follows a similar trend, stabilizing quickly but only slightly worsening over time, suggesting

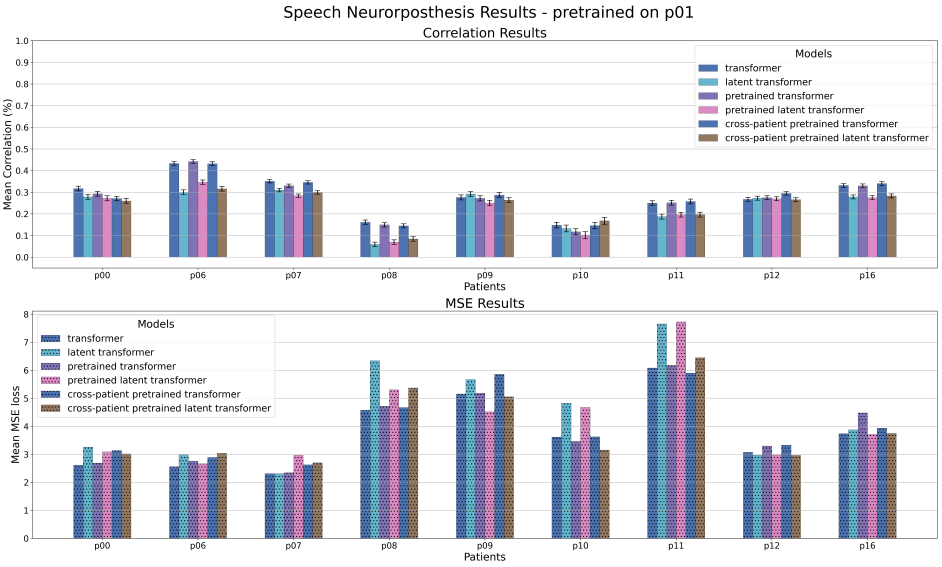


Figure 4.6: Comparison of model performance across different datasets using various models: **transformer**, **latent transformer**, **pre-trained transformer**, **pre-trained latent transformer**, **cross-patient pre-trained transformer**, **cross-patient pre-trained latent transformer**. The models were evaluated on a test set; top plot shows the correlation results, while the bottom plot shows the MSE results.

minimal degradation in model performance.

On the other hand, the latent cross-patient model exhibits a lower overall validation correlation, starting around 26%. After approximately 100 epochs, this correlation begins to decline, reaching around 20% by epoch 200. Inspecting patient-specific history reveals that the validation correlation declines across multiple patients, including 00, 07, and 09, indicating a significant drop in performance over time. Moreover, the validation MSE consistently worsens for all patients as training continues, further reinforcing the model's decreasing generalization capability.

Both models show clear signs of overfitting, as evidenced by the training metrics improving, even as the validation metrics become worse.

4.4 Transformer model speech neuroprosthesis capabilities

Figure 4.9 presents the results for two individual models trained on patient 01 data for either 500 or 5000 epochs. Both models achieve nearly identical Pearson correlation scores of approximately 46%. However, the model trained for 500 epochs yields a lower MSE of around 2.0, whereas the model trained for 5000 epochs exhibits a slightly higher MSE of around 2.5.

In Figures 4.11 and 4.12 the model's predictions are shown over the entire test set, either in the form of mel spectrograms or as an audio signal (synthesized by VocGan from the predicted mel spectrograms). These figures highlight the model's strong ability to predict speech activity from neural data, accurately distinguishing between regions of speech and silence.

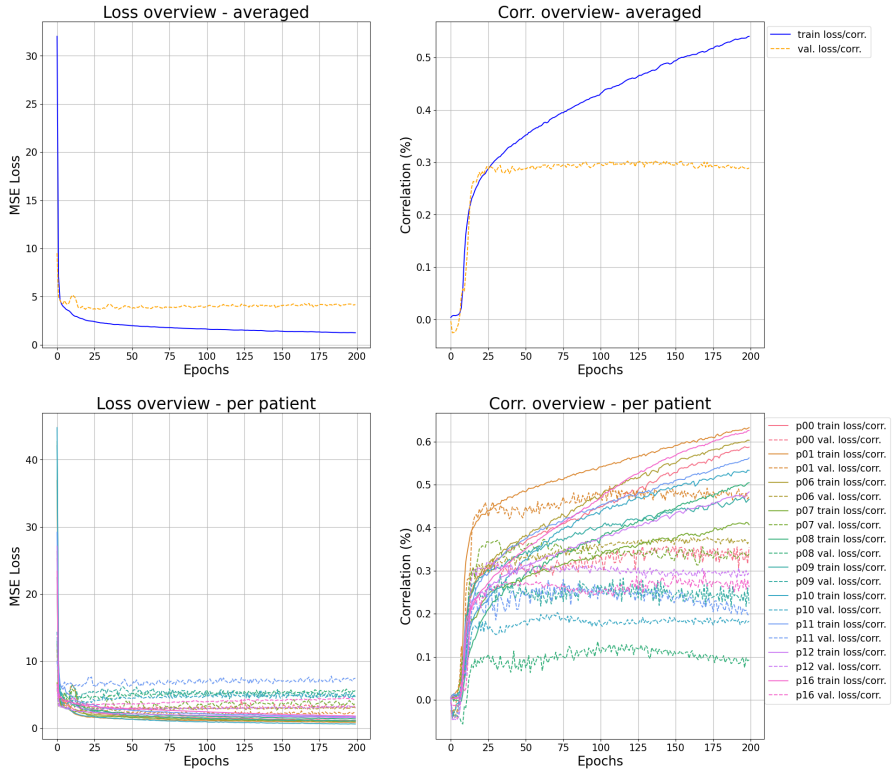


Figure 4.7: Training history of the normal cross-patient model. The model was trained and evaluated on datasets from 10 different patients. The top plots display the averaged training history across all patients, with the blue line representing the training metrics and the orange line indicating the validation metrics. The bottom plots illustrate the training history for each patient individually, where the solid lines represent the training performance and the dashed lines show the validation performance for each patient.

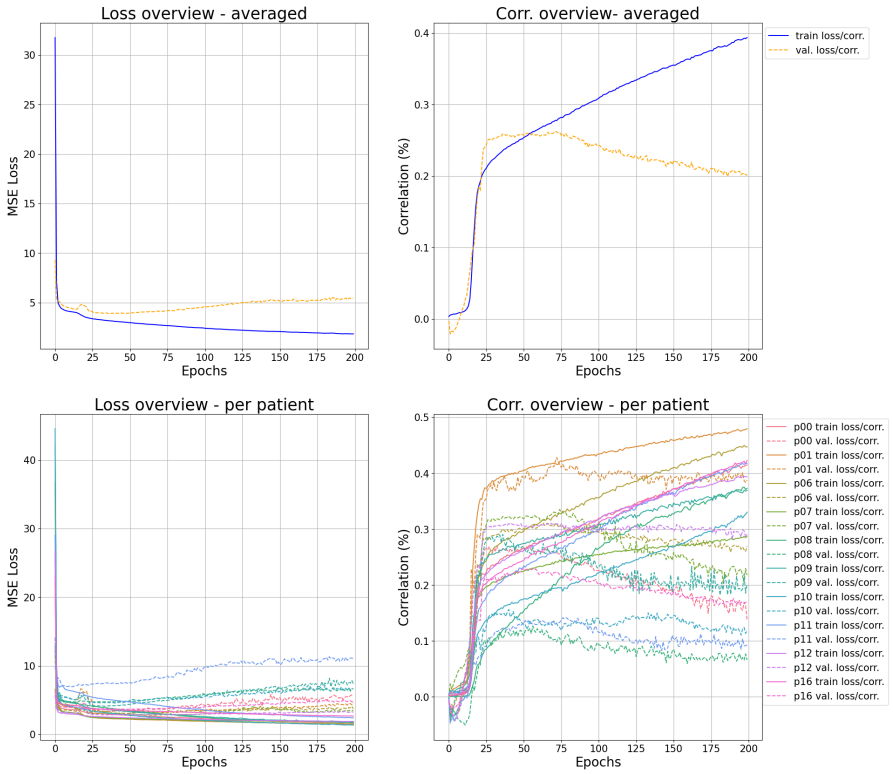


Figure 4.8: Training history of the cross-patient model trained on data encoded by autoencoders. The model was trained and evaluated on datasets from 10 different patients. The top plots display the averaged training history across all patients, with the blue line representing the training metrics and the orange line indicating the validation metrics. The bottom plots illustrate the training history for each patient individually, where the solid lines represent the training performance and the dashed lines show the validation performance for each patient.

Figure 4.10 shows the model's training history across three evaluation metrics: MSE, Pearson correlation, and MCD. All metrics exhibit a similar trend for both training and validation sets. Initially, there is a sharp improvement within the first 100 epochs. Afterwards, the validation metrics gradually plateau, while the training metrics continue to improve for approximately 3000 epochs, after which they also begin to plateau.

Figure 4.13 displays the mel-spectrogram reconstruction at different stages of training (500, 1000, 2500, and 5000 epochs), with the original mel-spectrogram shown in the top subplot. At 500 epochs, the model outputs constant energy across time for all spectral bins, showing no clear structure. By epoch 1000, the model begins to develop structure in the reconstructions, outputting varying energy levels over time for the spectral bins. This creates distinct spectral blocks that could resemble separate speech units. By epoch 2500, the reconstructions become more detailed, gradually resembling the structure of the original mel-spectrogram. At 5000 epochs, there is only marginal improvement over epoch 2500, but the reconstructions appear more detailed.

Figure 4.15 shows the average correlation across the spectral bins for the testing data. The first 35 bins (except for the first bin) maintain a relatively high and consistent correlation, around $47 - 49\%$. After this point, the correlation drops slightly by a few percentage points to approximately 44–46% over the next 20 bins. Starting from bin 55 to around bin 72, the correlation decreases more noticeably, reaching around 0.40. Finally, from bin 72 to the last bin, there is a steady decline in correlation, ending at around 30% for the last spectral bin.

Figure 4.14 shows the distribution of correlation values between predicted and original mel-spectrograms based on the percentage of speech content in the test windows. For windows that contain only silence, the correlation between predicted and original spectrograms is generally close to 0%, with occasional windows displaying higher correlation values. When the windows consist entirely of speech signals, the correlation centers around 17%. An interesting pattern emerges when the windows contain a mix of speech and silence. For windows with about 10% speech content, the correlation peaks at nearly 100%. However, as the percentage of speech within the window increases, the correlation steadily declines. In windows with approximately 90% speech, the correlation drops to around 50%.

Figure 4.16 illustrates the training history of an individual model trained on the patient 01 data with teacher forcing enabled. Initially, the validation correlation rises sharply to approximately 36%, stagnating briefly for a few hundred epochs before climbing again to nearly 40%. Following this phase, the training metrics stabilize for a few thousand epochs, up until around epoch 4000, where the validation correlation begins to decline steadily, eventually settling at around 25%. This decline is accompanied by noticeable instability in the validation correlation, which fluctuates within a wide range over short intervals, indicating unstable performance during certain phases of training. A similar trend is observed in the MCD metric. The MCD value decreases rapidly at the start, followed by a brief stagnation period, after which it continues to decrease until approximately epoch 1000. At this point, the validation MCD stabilizes before starting to increase at around epoch 4000. In both correlation and MCD metrics, there is clear overfitting, as evidenced by the significant difference between training and validation metrics.

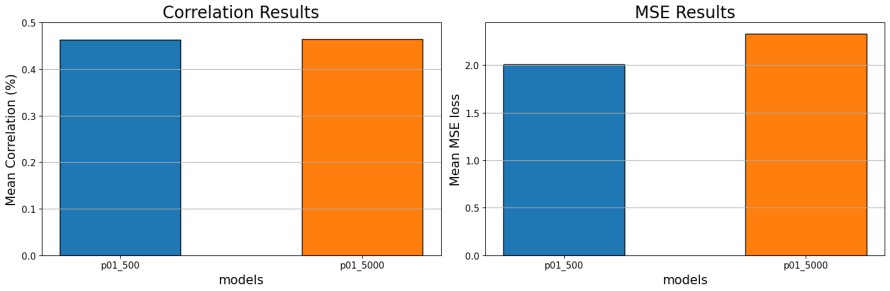


Figure 4.9: Pearson correlation and MSE results for two individual models trained on patient 01 data. The model labelled *p01_500* was trained for 500 epochs, while the model labelled *p01_5000* was trained for 5000 epochs.

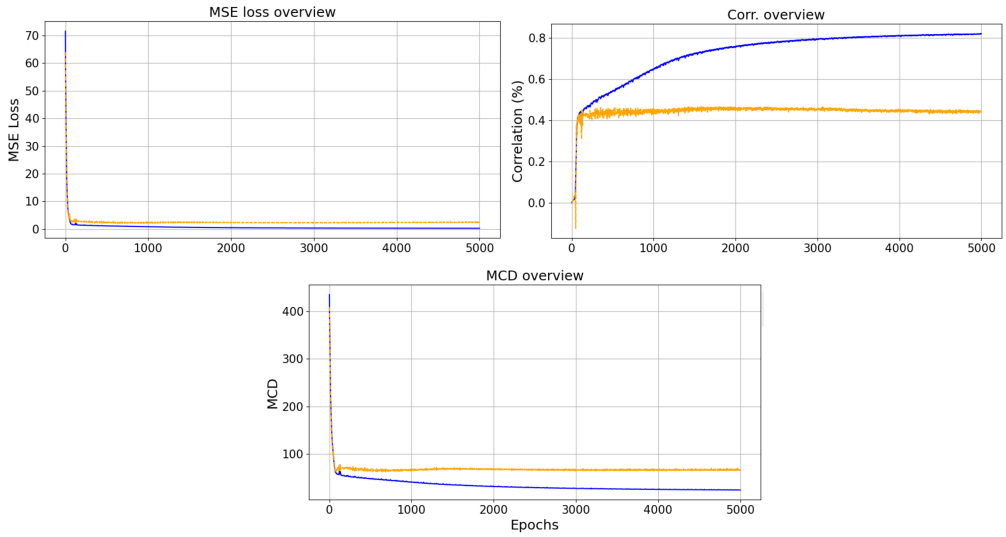


Figure 4.10: Training history for an individual model trained for 5000 epochs on patient 01 data. Each subplot represents the training history for a different metric. The blue line represents the training performance, and the orange line represents the validation performance. Top left: MSE loss; Top right: Pearson correlation; Bottom: MCD.

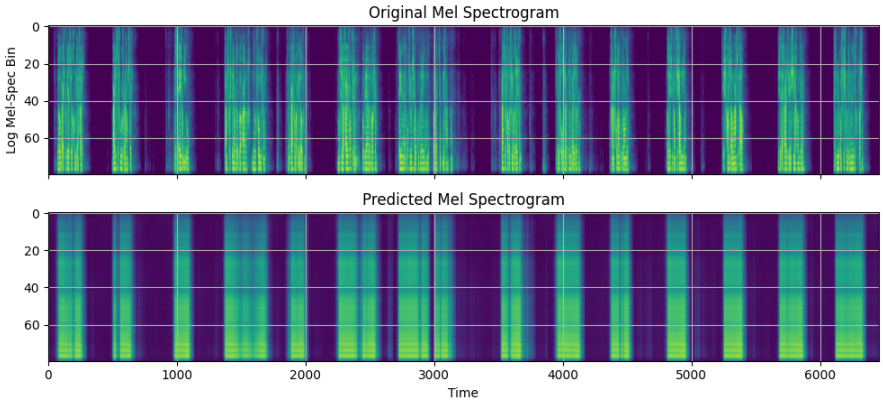


Figure 4.11: Mel-spectrogram predictions for patient 01 over the entire test set made by the individual model trained for 500 epochs. The model, trained to predict windowed data, was first evaluated on the testing set, and then the windowed predictions were stitched together to reconstruct the full test sequence. The top plot shows the ground truth mel-spectrogram, while the bottom plot displays the predicted mel-spectrogram.

4.5 Pre-trained model

Figures 4.17 and 4.18 show the training history for the individual and the pre-trained models using patient 06 data. The pre-trained model reaches a 30% correlation in approximately 25 epochs, compared to the 100 epochs required by the individual model for achieving the same correlation. Moreover, the pre-trained model achieves a higher validation correlation by the end of its training. Despite these differences in training speed and validation performance, the test set results for both models are nearly identical, with the pre-trained model showing a slight edge at 44% compared to 43% for the individual model.

Comparing the mel-spectrogram reconstructions between the two models 4.19 and 4.20 highlights that the pre-trained model produces more accurate reconstructions. The pre-trained model, rather than predicting a nearly constant energy flow across spectral bins, generates a more dynamic structure, with energy distributed more effectively across time. Although the predictions produced by the pre-trained model still lack finer details, they follow the original spectrogram more closely than the predictions made by the individual model, which struggles to capture the time-dependent variations in energy.

4.6 Latent space Visualization

Figure 4.21 shows the distribution of averaged embeddings for the datasets from different patients. Each subplot represents latent data encoded by a different model: subfigure (a) uses an autoencoder to encode the entire dataset, subfigure (b) shows the cross-patient model encoding only the test data, and subfigure (c) illustrates the latent cross-patient model also encoding only the test data. It is evident that data from different patients occupies distinct regions in the latent space, with no overlap. Furthermore, there is a noticeable separation between points with high speech content and those with little or no speech. This is particularly clear in the patient's 07 and 09 datasets in subfigure (a), where the points are grouped according to their speech content.

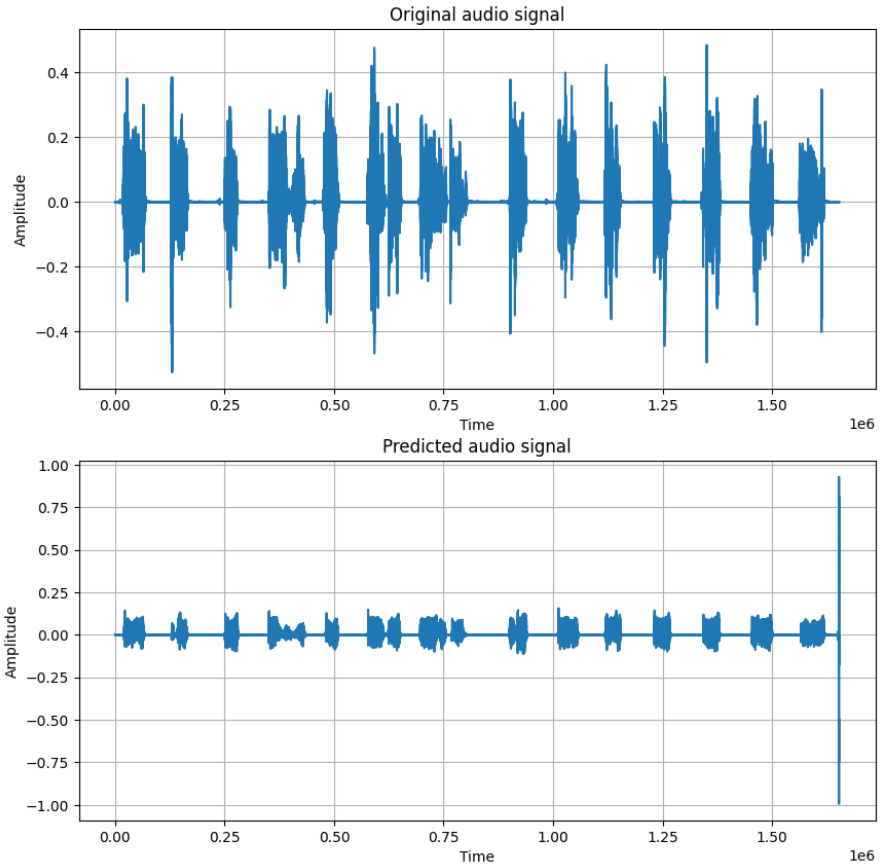


Figure 4.12: Synthesized audio signal from the predicted mel-spectrogram over the entire test set of patient 01. The audio was generated using the pre-trained VocGan model based on the predicted and stitched-together mel-spectrogram.

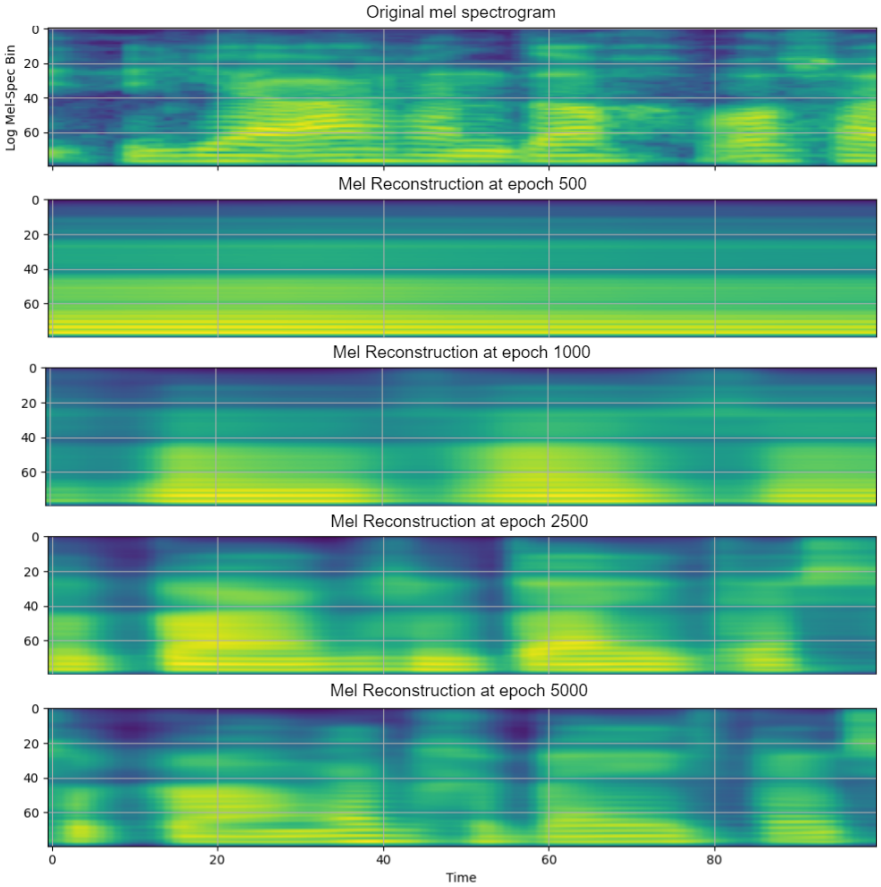


Figure 4.13: Mel-spectrogram predictions for the same test window made by the individual encoder-decoder model trained on patient 01 data. The top plot shows the original ground truth mel-spectrogram, while the plots below display the predictions at different stages of training: 500 epochs, 1000 epochs, 2500 epochs, and 5000 epochs.

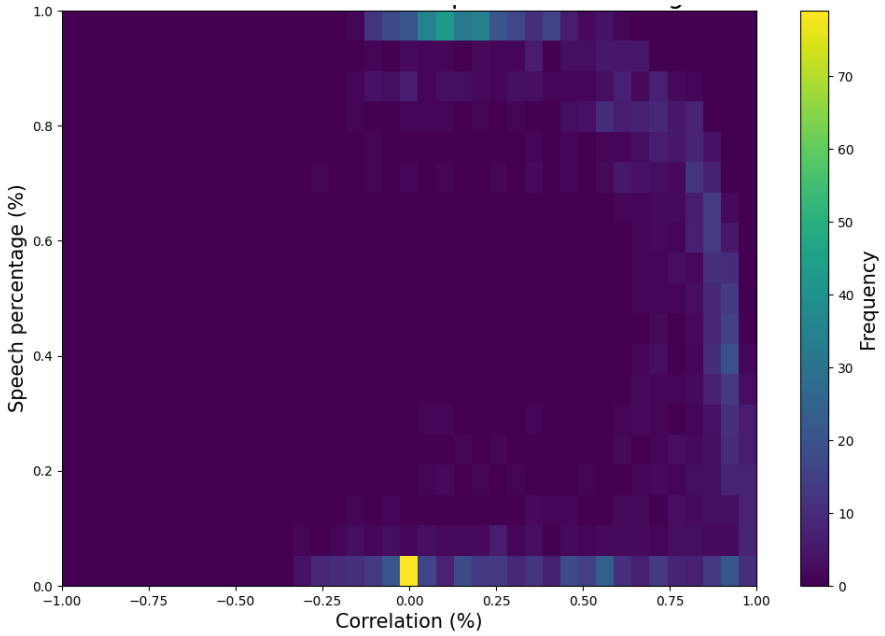


Figure 4.14: Heatmap illustrating the correlation across Patient 01 test set at varying speech percentages. For each window of the mel-spectrogram, the correlation and corresponding speech percentage were calculated.

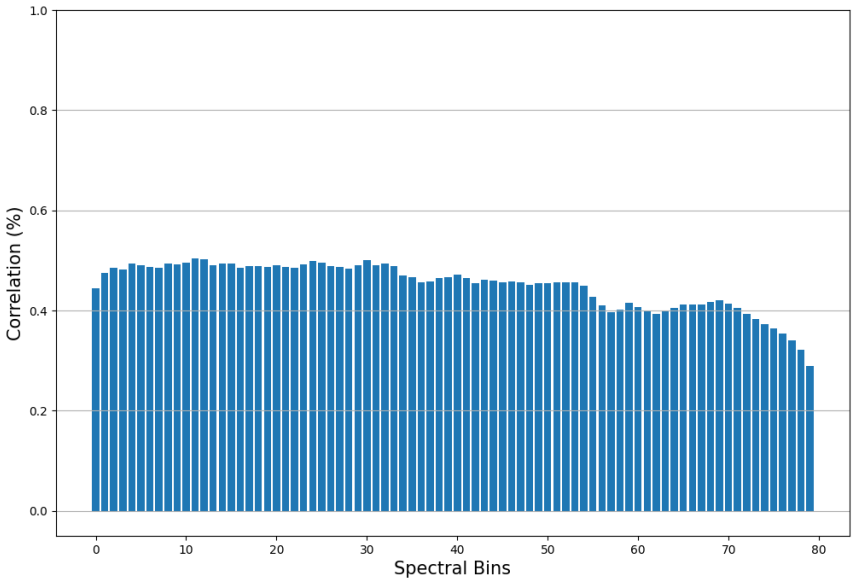


Figure 4.15: Average correlation calculated for each spectral bin over the test set of patient 01.

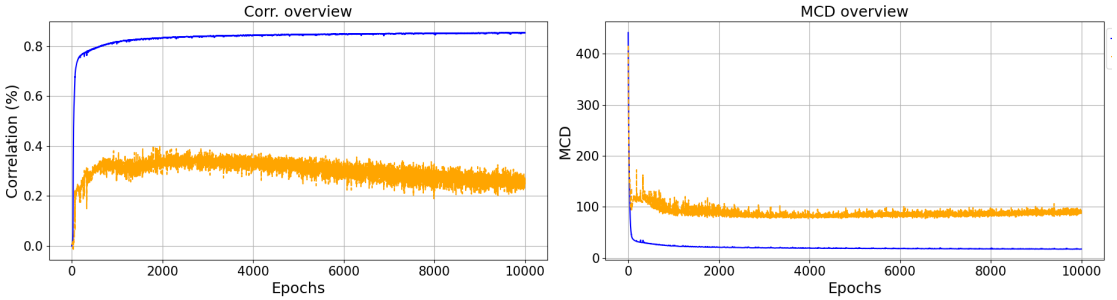


Figure 4.16: Training history for an individual model with teacher forcing enabled trained for 10000 epochs on patient 01 data. The blue line represents the training performance, and the orange line represents the validation performance. The left plot shows the training history for Pearson correlation while the right plots shows the training history for MCD.

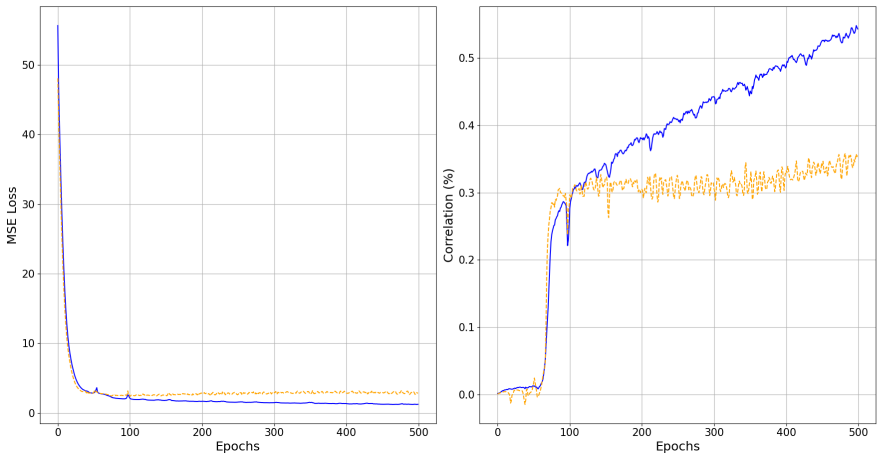


Figure 4.17: Training history for an individual model trained for 500 epochs on patient 06 data. The blue line represents the training performance, and the orange line represents the validation performance. The left plot shows the training history for MSE loss while the right plot shows the training history for Pearson correlation.

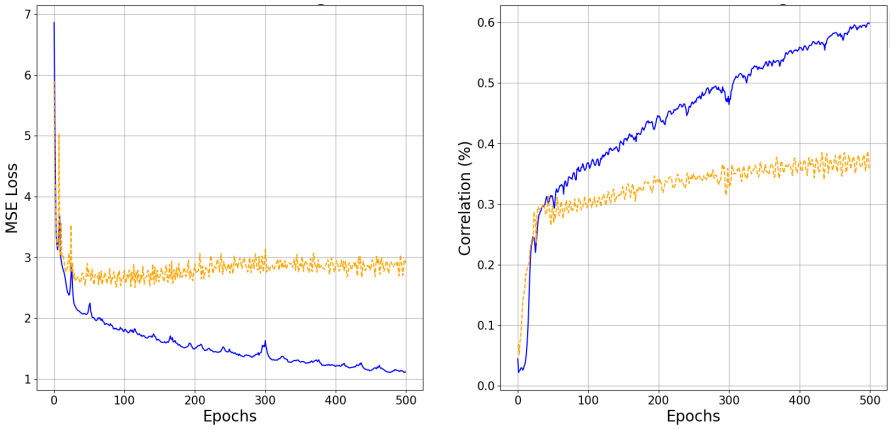


Figure 4.18: Training history for a pre-trained model, trained for 500 epochs using Patient 06 data. The pre-trained model was derived from a separate model, which was initially trained for 500 epochs on Patient 01 training data; the decoder from this model was then utilized for the pre-trained model. The blue line indicates the training performance, while the orange line represents the validation performance. The left plot displays the training history for MSE loss, and the right plot presents the training history for Pearson correlation.

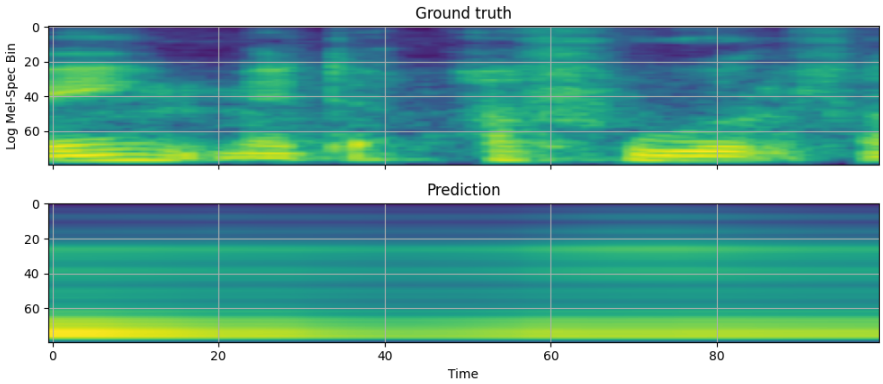


Figure 4.19: Mel-spectrogram predictions for a patient 06 test mel-spectrogram window. The prediction was made by an individual model trained for 500 epochs on patient 06 data. The top plot shows the ground truth mel-spectrogram, while the bottom plot shows the predicted mel-spectrogram.

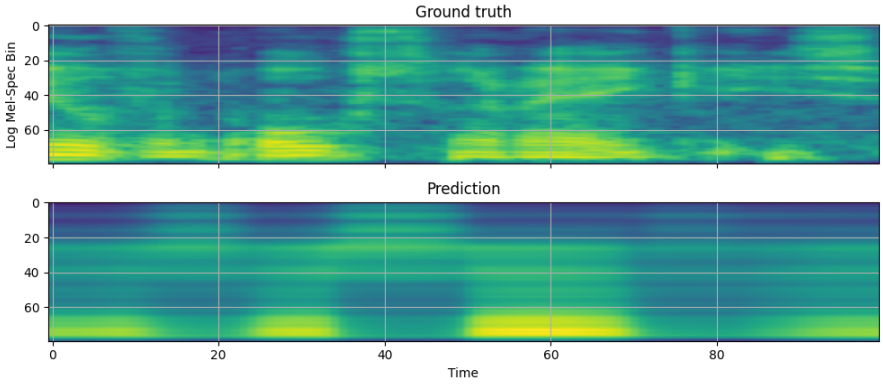
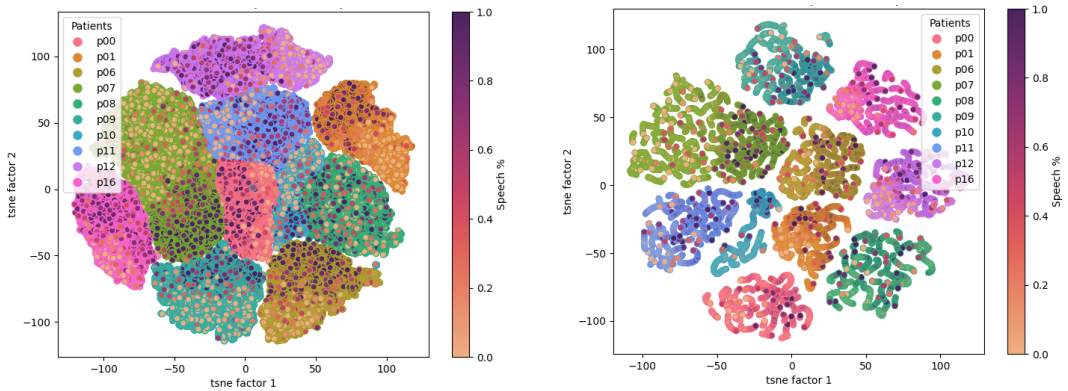


Figure 4.20: Mel-spectrogram predictions for a patient 06 test mel-spectrogram window. The prediction was made by a pre-trained model trained for 500 epochs on patient 06 data. The top plot shows the ground truth mel-spectrogram, while the bottom plot shows the predicted mel-spectrogram.

An exception is the dataset for patient 08, which shows no such distinction.

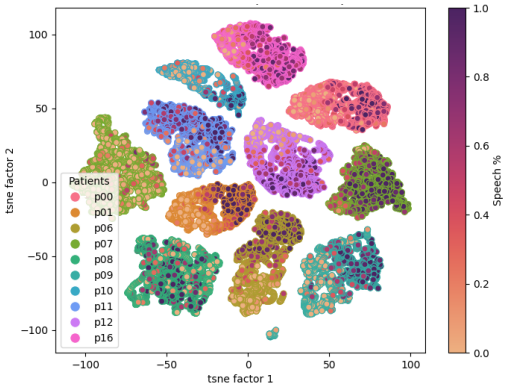
Figures 4.22 and 4.23 display the trajectory lines for the cross-patient model trained on latent and normal data, respectively, at different time points. Each subplot captures the distribution of points at a specific moment in time, where the color of the points corresponds to different patient datasets.

In both sets of figures, it is evident that as time progresses, data points from different patients do not overlap, highlighting the model's ability to separate patient-specific data in the latent space. However, a key difference can be observed in the dynamics of these points between the normal and latent cross-patient models. For the latent cross-patient model 4.22 the positions of data points shift considerably over time for most patient datasets, suggesting a more dynamic representation of speech activity. An exception to this trend is patient 08, whose points remain relatively stationary throughout the time course. By contrast, in the normal cross-patient model 4.23 the trajectories appear less fluid, with less pronounced shifts in the positioning of data points over time.



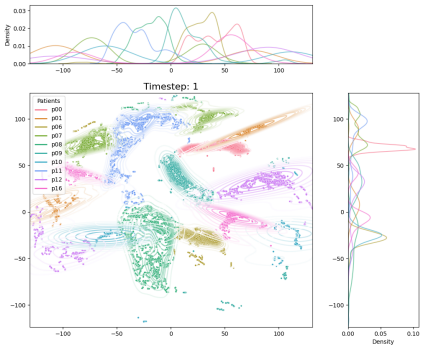
(a) Autoencoder latent data

(b) Cross-patient normal data

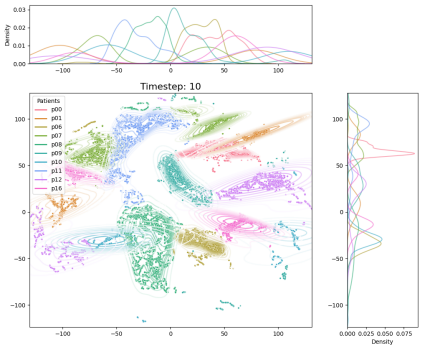


(c) Cross-patient Latent data

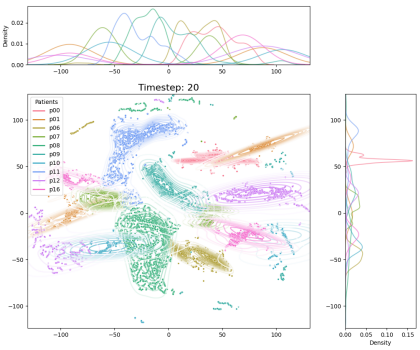
Figure 4.21: t-SNE representation of the latent space for different models, where each point's temporal dimension has been averaged and its feature set reduced to a 2D space. Points are color-coded by speech percentage. Each plot represents latent data extracted by a different model configuration (a) Latent data encoded by an autoencoder model (including points from the entire dataset), (b) latent data encoded by a cross-patient transformer, (c) latent data encoded by a cross-patient transformer model which was trained on data encoded by an autoencoder.



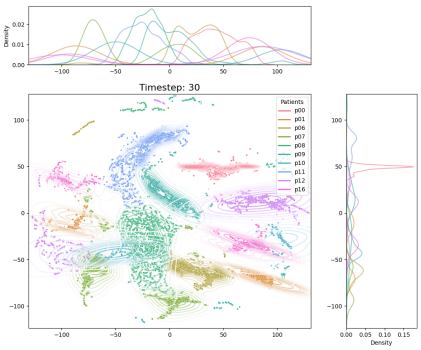
(a) t=0



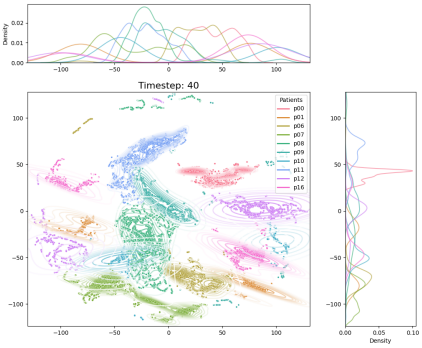
(b) t=10



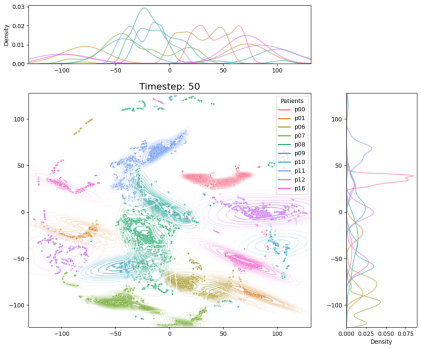
(c) t=20



(d) t=30

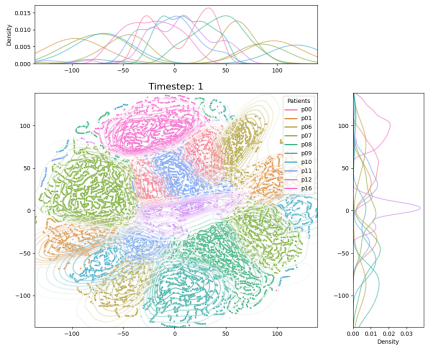


(e) t=40

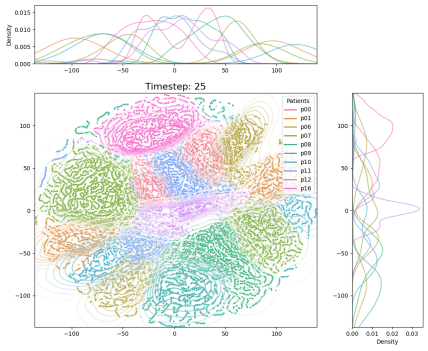


(f) t=50

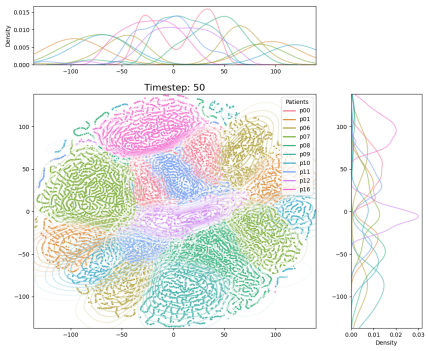
Figure 4.22: Latent data extracted by the latent cross-patient model: t-SNE representation of trajectory lines, illustrating the distribution of testing points across patients at various time points. Each main plot is accompanied by two subplots—one on the top and one on the right—showing the distribution of t-SNE feature 1 and feature 2, respectively. The contour lines found on the main plots provide a visual representation of the density and structure of the underlying point distribution at each specified time point.



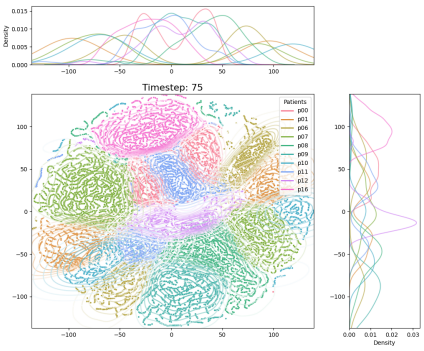
(a) t=0



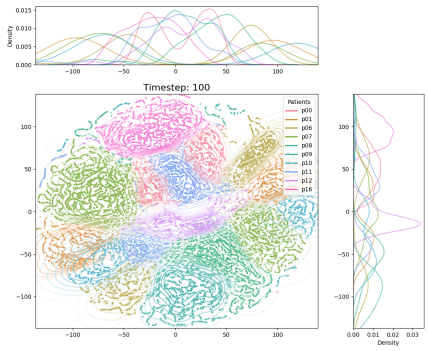
(b) t=25



(c) t=50



(d) t=75



(e) t=100

Figure 4.23: Latent data extracted by the normal cross-patient model: t-SNE representation of trajectory lines, illustrating the distribution of testing points across patients at various time points. Each main plot is accompanied by two subplots—one on the top and one on the right—showing the distribution of t-SNE feature 1 and feature 2, respectively. The contour lines found on the main plots provide a visual representation of the density and structure of the underlying point distribution at each specified time point.

Chapter 5

Discussion and Conclusion

5.1 Autoencoder reconstructions and latent space

From the results presented in 4.1 a difference of just 0.4 MSE is enough to differentiate an autoencoder capable of accurate reconstruction from one that struggles with generalization. Interestingly, this performance gap in reconstruction does not necessarily imply that the latent space lacks sufficient information for speech synthesis.

As shown in 4.4 (b), it can be observed that for patient 01, the performance of the LR model using latent space features is nearly identical to that of the input data, suggesting that the key features required for speech neuroprosthesis remain well-preserved in the latent representation. Moreover, the model's performance on the reconstructed data exceeds that of the baseline.

This behavior is rather unexpected, as the purpose of the autoencoder in this research is to compress input data into a latent space, and due to that the reconstructed data should be less informative than the original data for speech neuroprosthesis tasks. However, in this case, the latent space matches the performance of the baseline, while the reconstruction surpasses it. While the reconstructed data might contain representations that are better suited for speech neuroprosthesis, more plausibly, it is the flaw of the model itself, as LR is not the most optimal model for speech neuroprosthesis.

Conversely, as shown in Figure 4.4 (a) shows that the LR model trained on patient 00 data performs worse than the one trained on patient 01 data, and that the latent space representation for patient 00 yields poorer results compared to the input data. This outcome is expected, as the latent space represents a compressed version of the input, inherently containing less information.

Nonetheless, it's important to recognize that, despite the patient 01 autoencoder producing less accurate reconstructions and higher MSE values compared to the patient 00 model, the neural signals in patient 01's data seem to be more informative for speech neuroprosthesis. This ultimately leads to better performance in the speech neuroprosthesis task when using an LR model, even though the reconstruction quality itself is lower.

5.2 Speech Neuroprosthesis Results

Overall, all tested models suffer from overfitting issues and their predictive performance can still be improved as 500 epochs or less (depending on the model) is not enough to reach a point where the model does not learn anymore. 500 epochs is however enough to allow the model to distinguish between speech and silence regions. This model capability is most probably represented by the sudden rise in validation correlation observed on the training plots (Figures 4.3, 4.10 and 4.18).

Differences in results between latent and normal models One of the main reasons why most of the latent models perform worse is the fact that the autoencoder compresses information and therefore, the data carries less information in it. Additional data is lost due to the model performance. As seen by Figure 4.1 most of the autoencoders have relatively high MSE which makes the features found in the latent space carry even less information that would be useful for speech synthesis.

Differences in results between patients The discrepancy between results achieved on different patient datasets can be attributed to the quality of data and the brain regions from which the neural activity was recorded. For example, as shown in Figure 3.1, patient 08 had only electrodes implanted in the right hemisphere, while patient 01 had electrodes implanted in both hemispheres. It is also possible that during the data acquisition process, faulty equipment or some other error corrupted the recorded data.

5.2.1 Speech synthesis and evaluation metrics

Pearson Correlation was employed as one of the key evaluation metrics for assessing the quality of the reconstructed speech mel-spectrograms. While it offers a broad indication of the reconstruction quality, it does have certain limitations. The majority of models in this study were trained for 500 epochs or fewer, due to the mentioned tendency for overfitting. In Figure 4.10 around epoch 300, the validation MSE and correlation metrics plateau, whereas the training metrics continue to improve. This would indicate that extending the training duration beyond this point does not yield meaningful performance improvements. The same figure also shows, that even when the training continued there was no significant improvement over the validation metrics. The comparison shown in Figure 4.9 further supports this observation - while the model trained for 5000 epochs achieves the same Pearson correlation, it actually scores a worse MSE.

However, a more detailed examination of the predictions made at various stages of model training uncovers a nuance: despite similar validation metrics, the quality of the predictions made at epochs 500 and 5000 can differ substantially. This suggests that validation metrics alone may not fully capture the progression in prediction quality during extended training. Additionally, although speech intelligibility remains low — making it difficult to discern what a patient is saying — inspecting the audio clips (which can be found at the: Master Thesis Repository) reconstructed from the predicted mel-spectrograms reveals the model's ability to capture and replicate the distinct voice characteristics of the patient. This capability highlights that, while the intelligibility of the speech may be limited, the model successfully retains essential

aspects of the patient's vocal identity, which is a critical feature for speech neuroprosthesis applications.

The flaw of Pearson correlation is further highlighted by Figure 4.14. Testing windows that contain exclusively speech or silence yields lower correlation results, whereas windows with a mix of both tend to score higher. This outcome might appear contradictory when compared to the stitched predictions over the entire test set (Figure 4.11), which demonstrates that the model can generally distinguish between speech and silence regions. Therefore the correlation for windows containing only silence should be reasonably high.

However, even a small misplaced prediction within a testing window can significantly disrupt correlation scores leading to low correlation even in windows where the model identifies silence accurately. For example, if the model predicted for 5 out of 100 time points an energy output, the correlation would drop. As soon as speech is introduced to a testing example, those false positives (predicting speech, where there should be silence) are almost non-existent. The model detects this activity and often pinpoints it accurately in time. For testing windows containing mostly silence although the model's reconstruction of speech is suboptimal, the dominant silence signal, which is predicted with high precision results in a high correlation score. As the proportion of speech in the window increases, however, the reconstruction errors from poorly synthesized speech begin to overshadow silence predictions, thus lowering the overall correlation. Annotated predictions illustrating this issue are provided in Appendix A.3.

Unfortunately, using MCD for progress tracking does not help in model evaluation, as this metric suffers from the same issue as person correlation. This presents a challenge in accurately assessing the full potential of the encoder-decoder transformer model, as the standard metrics do not fully capture the model's learning capability.

5.2.2 Cross-patient model training

The architecture of the cross-patient network allows its decoder to essentially train on a larger dataset, as it receives training examples from multiple patients. While increasing the amount of training data is often seen as a solution to overcome overfitting[46], the cross-patient model still suffers from this issue. Additionally, despite having access to more data, the model does not seem to benefit from it when compared to the performance of individual models. This larger training dataset, however, likely contributes to the model learning faster initially compared to individual models. It accelerates its early training phase, but long-term improvements remain unclear, as the model has only been trained for 200 epochs so far.

At its current level of training the cross-patient model is capable of detecting speech activity across different patients but it's not capable of capturing the general shape of the mel-spectrograms (see Appendix A.5). Listening to the reconstructed audio clips suggests that the model may be capturing some aspects of each patient's unique voice. This raises the possibility that the model could eventually synthesize speech for individual patients without compromising between different datasets. However, further testing through extended training is necessary to fully validate this potential.

The main limitation of the cross-patient model is its significantly slower training speed. When trained on data from 10 patients, the model is approximately 10 times slower than a model trained

on data from just one patient. This slower pace presents a challenge for extended training, which is crucial for assessing the long-term benefits of the architecture. Given the training speed, it might be more practical to train individual models for each patient rather than relying on a single cross-patient model.

5.2.3 Pre-trained model training

The pre-trained model appears to benefit from a decoder that had been previously trained, despite the fact that the training data for this decoder originated from a different patient. Interestingly, the predictions made by the pre-trained model exhibit characteristics similar to those produced by the individual model trained for 1000 epochs. At this stage, the decoder seems to capture some voice characteristics of patient 06. However, upon inspecting the reconstructed audio clips, it becomes clear that it is still too early to make a definitive judgment about the quality of the predictions. Overall the predictions seem consistent with the total amount of training the decoder in the pre-trained model received, as it was initially trained for 500 epochs and then for an additional 500 epochs.

However, it remains unclear whether this advantage stems from the interchangeability of decoders between different models or if it is simply due to the initial 500 epochs of training, which are sufficient to train the decoder to distinguish between speech and silence regions. The subsequent training then likely built upon this foundational understanding. An intriguing question arises as to whether the decoder's transferability would diminish with longer initial training. For instance, had the decoder been trained for 5000 epochs before being transferred, it might have struggled more to adapt to a new patient's data, requiring significantly more re-tuning to achieve comparable results.

5.2.4 Teacher forcing

The downsides of teacher forcing are well-documented in the literature, as models employing this technique can suffer from exposure bias[23]. This bias occurs because the model is trained on ground truth sequences rather than its own predictions, leading to inconsistencies between the training and inference phases. The individual model trained with teacher forcing in this work is no exception, yielding lower overall results compared to models trained without it. The faster plateau in training correlation further emphasizes this issue, as it indicates a disconnect between training performance and real-world predictions during inference.

To address the limitations of teacher forcing, alternative approaches like Professor Forcing[47] and Scheduled Sampling[48] have been proposed. These techniques aim to reduce exposure bias by gradually allowing the model to predict based on its own outputs. Additionally, transformer models can also benefit from such methods as illustrated in the following study[49].

5.2.5 Model performance compared to relevant studies

Directly comparing the model developed in this research to other decoding models is challenging, as the datasets used across studies lack standardization and variations in quantitative evaluation methods can affect the reported results. For example, the results shown in Figure 4.5 were obtained by calculating either Pearson correlation or MSE on each windowed prediction within the test set and then averaging those metrics. However, when these same windowed predictions are stitched together to form a continuous mel-spectrogram of the entire test set (as illustrated

in Figure 4.11), the Pearson correlation and MSE results differ (see Appendix A.7).

Nevertheless, a possible comparison can be made with the work of Kohler et al. [25], as they used the same dataset. In this work, the best-performing model achieved an MSE of around 1.4 and a correlation of about 0.75%. In comparison, the individual model trained for 5000 epochs on patient 01 data, scored a worse MSE of around 2.3 and a correlation of 46%. Listening to the reconstructed audio at Jonas Kohler research repository (p3 audio files) and at Master Thesis repository demonstrates that both models capture some aspects of the patient's voice characteristics, although intelligibility remains limited in both reconstructions.

While Chen et al.[29] employed the same transformer architecture to synthesize single words from sEEG data, they did not provide audio reconstructions, leaving the clarity of their results ambiguous. However, they reported a lack of intelligibility in the reconstructed audio waveforms, though they observed that the model could reliably distinguish between speech and silence. This observation aligns with the results of this study.

5.3 Alignment within the latent space

There is no overlap between different patient datasets for any of the presented latent space extraction methods. While both the autoencoder and encoder-decoder transform neural data into the same latent space, each patient essentially occupies a distinct region within this space. This suggests that the latent space features do not represent the same aspects of the input data across different patients and that the current models (i.e. autoencoder and encoder-decoder) are insufficient to enforce alignment between differently distributed datasets. The absence of alignment is further supported by the observation that embeddings for points with different speech content (such as speech or silence) occupy distinct spaces, even though they describe the same underlying characteristic.

Despite the lack of overlap between patients, a notable difference emerges when examining the trajectory lines of embeddings within the latent space. The behavior observed in latent trajectory lines (as seen in 4.22) might initially imply that using an autoencoder for feature extraction helps the model capture temporal dynamics in the neural activity. However, when the trajectory lines are examined more closely within the autoencoder's latent space (see Appendix A.2), no such temporal pattern emerges. Instead, this temporal progression is more likely to originate from the combination of the autoencoder's feature extraction and the temporal encodings provided by the transformer model.

5.4 Future outlook

5.4.1 Evaluation metrics

One of the main challenges in this research was the use of evaluation metrics that did not fully reflect the performance of the speech synthesis model. To address this in future work, incorporating additional evaluation metrics into the training process is recommended. One promising metric is the Short-Time Objective Intelligibility (STOI) measure [50], which has been successfully utilized by Chen et al.[38] to evaluate speech intelligibility. Adopting STOI could provide a more comprehensive assessment of the model's ability to capture key speech characteristics.

5.4.2 Multiclass classification

One of the primary challenges for the proposed transformer model lies in its difficulty in generating intelligible speech. Research by Metzger et al. [9] demonstrated an effective approach by using HuBERT to convert raw audio into discrete speech units. Instead of directly predicting mel-spectrograms, their model first predicts these discrete units, which are then used by a pre-trained model to generate high-quality mel-spectrograms. This adjustment simplifies the problem from a regression task to a multi-class classification problem, potentially improving the transformer model's ability to produce more intelligible speech. While Metzger's et al. approach utilized an RNN to predict speech units, it would be valuable to explore how a transformer-based model compares in terms of performance and speech quality.

5.4.3 Manifold Alignment

As mentioned in the introduction, projecting two datasets, A and B, into the same dimension can be achieved through manifold alignment, where the core idea is to transform the features of each dataset through specific mappings based on points of similar local geometry. This results in a common space where the structural relationships within each dataset are preserved[13].

An illustrative example of manifold alignment can be found in the work of Lejeune and Driessens [51], who applied this technique to align feature sets in the MNIST dataset. In their study, each digit (e.g., the numbers 3 and 4) was treated as a separate dataset, with each number represented by a 28×28 pixel grayscale image. These images, though sharing the same dimensionality, differ in pixel distribution due to the different handwritten forms of each digit. To align these datasets, new feature sets were first constructed by extracting latent spaces using a deep belief network, and then the datasets were aligned using manifold alignment. This allowed the creation of an overlapping set of features for datasets described by a different underlying distribution.

5.4.4 Deep Metric Learning

Different studies mentioned in section 2.2 have employed deep metric learning with contrastive loss to create a unified latent space for different datasets[37][33][35]. However, these studies primarily addressed classification problems, where contrastive loss operates on distinct classes to change the alignment of the underlying data. Therefore for speech neuroprosthesis, the first step would be to discretize the audio signal into classes or speech units. Metzger et al. [9] already demonstrated that it is feasible to transform speech data into speech units with HuBERT and perform successful speech neuroprosthesis with it.

With discretized speech units and contrastive loss forming one part of the solution, the next challenge lies in projecting datasets with different channel counts into a unified latent space. This step could involve adapting the method proposed by Chen et al. [38], where tokens represent individual channels at different time points instead of tokens representing all channels at different time points. This would enable the model to process input data with varying channel counts, leveraging positional bias to account for the location of recording electrodes.

Finally, the proposed pipeline would be similar to that of Guetschel et al. [33], featuring a distinct embedding function for aligning data from each patient, alongside a separate classification model that synthesizes data from the generated embeddings. The embedding function would

utilize a transformer encoder capable of processing input with an arbitrary number of channels. This encoder would transform the data using contrastive loss derived from a multi-class classification problem based on speech units obtained from HuBERT.

5.4.5 Neural data embeddings

An unexplored direction in the field is the development of task-independent neural embeddings. Presently, studies on neural embeddings or cross-patient dataset alignment generally focus on specific tasks. In contrast, word2vec embeddings, though generated through a classification task, capture relationships solely between words rather than between words and another modality. This task independence adds versatility to word embeddings, broadening their applicability across diverse contexts.

However, EEG embeddings' reliance on specific tasks—such as classification—limits their generalizability, resulting in a fragmented research landscape centred around individual tasks, such as motor or emotion classification. While this task specificity is not inherently problematic, it constrains the broader applicability of EEG embeddings and hinders the potential for cross-task insights.

5.5 Research Questions - revisited

- **Is it possible to align different patient datasets in the latent space using a combination of convolutional auto-encoders and a transformer model?**

  The results presented in section 4.6 indicate that encoding input data into a latent space using an autoencoder, followed by a cross-patient model, does not achieve effective alignment across different patient datasets. Similarly, using the autoencoder and transformer models independently also fails to align datasets from different patients. Each patient's data occupies a distinct region within the latent space, with no apparent overlap, suggesting that each dataset is characterized by a unique set of features. This outcome implies that an indirect alignment approach is insufficient to ensure feature alignment across patients.

- **Is an encoder-decoder transformer model capable of efficient speech neuroprosthesis?**

  Overall, the results presented in this work suggest that the encoder-decoder transformer model may hold potential for speech neuroprosthesis applications but at the current stage it is not capable of efficient speech neuroprosthesis.

  After the training process reaches a plateau, the model primarily focuses on learning the temporal dimension, successfully predicting when speech and silence occur but struggling to accurately reconstruct speech. However, as discussed in Section 5.2.1, the model's apparent overfitting—indicated by discrepancies between training and validation metrics—does not diminish the quality of its predictions; rather, it highlights the model's ability to capture certain voice characteristics of the patients, ultimately improving its predictive accuracy. Consequently, although the highest correlation achieved in the best-case scenario is approximately 47%, the encoder-decoder transformer does possess the potential for accurate

speech reconstruction.

Moving forward, it is essential to introduce new evaluation metrics that better capture the model's learning process and to design architectures more optimized for speech neuroprosthesis. This could involve incorporating pre- or post-nets, as demonstrated by Jonas et al. [25] and Li et al. [14], or transform the problem into a multi-class classification problem by using quantized HuBERT speech units as targets. Additionally, since the transformer model trained with teacher forcing showed promise in reconstructing speech and significantly accelerated training, adopting this training method could be a key step forward, enabling more experimentation and possible improvements.

- **Is there any improvement gained from training an sEEG-to-spectrogram model on multiple datasets?**

  The results presented in Sections 4.2 and 4.3 indicate that the cross-patient model performs comparably to the individual variants. Unfortunately, using multiple datasets does not prevent the model from overfitting, and there appears to be no clear advantage to using data from more than one patient. It is not clear what are the long-term effects of the many-encoders-to-one-decoder architecture. The model could balance the characteristics of different patients, limiting its ability to capture the unique voice features of any single patient or present bias towards a specific patient given how smaller datasets can be presented multiple times during a single training iteration. To confirm these findings, further research with longer training periods and alternative architectures better suited for speech neuroprosthesis would be necessary.

  In contrast, results presented in Section 4.5 demonstrate that pre-training a model on one patient and subsequently reusing its decoder for another patient can enhance the model's ability to capture voice characteristics for other patients. This suggests that pre-training encoder-decoder models could be a promising direction for future research.

5.6 Conclusion

The alignment of neural data across patients has been one of the central goals of this study. However, the current model architecture, which includes multiple encoders and a single decoder, does not ensure that embeddings are shareable between different patients. Future work might explore deep metric learning, which has shown promise in generalizing across EEG datasets from different individuals, or explore different model architectures that are capable of generalizing across different patient datasets.

For speech neuroprosthesis applications, although accurately measuring the predictive performance of the encoder-decoder transformer model has been challenging, the results indicate promising potential for this specific model architecture. The model can effectively use sEEG data to distinguish between speech and silence, and with further training, it has shown the ability to synthesize speech, which while being unintelligible, reflects the vocal characteristics of individual patients.

The cross-patient model does not provide a significant advantage over patient-specific models. Using data from multiple patients simultaneously neither prevents overfitting nor enhances

the model's overall performance. Additionally, the cross-patient model fails to align data from different patients within a single latent space, indicating limitations in creating consistent neural embeddings for speech neuroprosthesis.

On the other hand, the pre-trained model demonstrates the potential for leveraging knowledge transfer by utilizing a trained decoder on different patient datasets. This finding suggests that further research into pre-trained architectures could be valuable, particularly in improving cross-patient generalization and embedding consistency for speech neuroprosthesis applications.

Overall, this research explored the applications of the encoder-decoder transformer model for speech neuroprosthesis, providing new valuable insights for this type of neural network architecture.

Bibliography

[1] Dennis J. McFarland and Jonathan R. Wolpaw. "Brain-computer interfaces for communication and control". In: *Commun. ACM* 54.5 (May 2011), pp. 60–66. ISSN: 0001-0782. DOI: 10.1145/1941487.1941506. URL: https://doi.org/10.1145/1941487.1941506.

[2] Susan Fager et al. "New and emerging access technologies for adults with complex communication needs and severe motor impairments: State of the science". In: *Augmentative and Alternative Communication* 35 (Jan. 2019), pp. 1–13. DOI: 10.1080/07434618.2018.1556730.

[3] Ujwal Chaudhary, Niels Birbaumer, and Ander Ramos-Murguialday. "Brain–computer interfaces for communication and rehabilitation". In: *Nature Reviews Neurology* 12.9 (2016), pp. 513–525.

[4] Alexander Silva et al. "The speech neuroprosthesis". In: *Nature Reviews Neuroscience* 25 (May 2024). DOI: 10.1038/s41583-024-00819-9.

[5] Maxime Verwoert et al. "Whole-brain dynamics of articulatory, acoustic and semantic speech representations". In: *bioRxiv* (2024). DOI: 10.1101/2024.08.15.608082. eprint: https://www.biorxiv.org/content/early/2024/08/15/2024.08.15.608082.full.pdf. URL: https://www.biorxiv.org/content/early/2024/08/15/2024.08.15.608082.

[6] Gerwin Schalk and Eric C. Leuthardt. "Brain-Computer Interfaces Using Electrocorticographic Signals". In: *IEEE Reviews in Biomedical Engineering* 4 (2011), pp. 140–154. URL: https://api.semanticscholar.org/CorpusID:206522826.

[7] Christian Herff, Dean J. Krusienski, and Pieter Kubben. "The Potential of Stereotactic-EEG for Brain-Computer Interfaces: Current Progress and Future Directions". In: *Frontiers in Neuroscience* 14 (2020). ISSN: 1662-453X. DOI: 10.3389/fnins.2020.00123. URL: https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2020.00123.

[8] Maitreyee Wairagkar et al. "An instantaneous voice synthesis neuroprosthesis". In: *bioRxiv* (2024). DOI: 10.1101/2024.08.14.607690. eprint: https://www.biorxiv.org/content/early/2024/08/19/2024.08.14.607690.full.pdf. URL: https://www.biorxiv.org/content/early/2024/08/19/2024.08.14.607690.

[9] Sean Metzger et al. "A high-performance neuroprosthesis for speech decoding and avatar control". In: *Nature* 620 (Aug. 2023). DOI: 10.1038/s41586-023-06443-4.

[10] CJ van Stam and ECW Van Straaten. "The organization of physiological brain networks". In: *Clinical neurophysiology* 123.6 (2012), pp. 1067–1087.

[11] Bryan Kolb and Ian Whishaw. "Brain Plasticity and Behavior". In: *Annual review of psychology* 49 (Feb. 1998), pp. 43–64. DOI: 10.1146/annurev.psych.49.1.43.

[12] Aleksandra Kawala-Sterniuk et al. “Summary of over Fifty Years with Brain-Computer Interfaces—A Review”. In: *Brain Sciences* 11.1 (2021). ISSN: 2076-3425. DOI: 10.3390/brainsci11010043. URL: https://www.mdpi.com/2076-3425/11/1/43.

[13] Chang Wang and Sridhar Mahadevan. “Manifold alignment without correspondence”. In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. IJCAI’09. Pasadena, California, USA: Morgan Kaufmann Publishers Inc., 2009, pp. 1273–1278.

[14] Naihan Li et al. “Neural speech synthesis with transformer network”. In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI’19/IAAI’19/EAAI’19. Honolulu, Hawaii, USA: AAAI Press, 2019. ISBN: 978-1-57735-809-1. DOI: 10.1609/aaai.v33i01.33016706. URL: https://doi.org/10.1609/aaai.v33i01.33016706.

[15] Yongqiang Wang et al. “Transformer-Based Acoustic Modeling for Hybrid Speech Recognition.” In: *ICASSP*. IEEE, 2020, pp. 6874–6878. ISBN: 978-1-5090-6631-5. URL: http://dblp.uni-trier.de/db/conf/icassp/icassp2020.html#WangMLLXMHTZZFZ20.

[16] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. “Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives”. In: *CoRR* abs/1206.5538 (2012). arXiv: 1206.5538. URL: http://arxiv.org/abs/1206.5538.

[17] Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *CoRR* abs/1301.3781 (2013). URL: http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781.

[18] Kaya and Bilge. “Deep Metric Learning: A Survey”. In: *Symmetry* 11.9 (Aug. 2019), p. 1066. DOI: 10.3390/sym11091066. URL: https://doi.org/10.3390%2Fsym11091066.

[19] Dor Bank, Noam Koenigstein, and Raja Giryes. “Autoencoders”. In: *CoRR* abs/2003.05991 (2020). arXiv: 2003.05991. URL: https://arxiv.org/abs/2003.05991.

[20] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[21] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014).

[22] Ronald J. Williams and David Zipser. “A Learning Algorithm for Continually Running Fully Recurrent Neural Networks”. In: *Neural Computation* 1 (1989), pp. 270–280. URL: https://api.semanticscholar.org/CorpusID:14711886.

[23] Marc’Aurelio Ranzato et al. “Sequence Level Training with Recurrent Neural Networks”. In: *CoRR* abs/1511.06732 (2015). URL: https://api.semanticscholar.org/CorpusID:7147309.

[24] Gopala K. Anumanchipalli, Josh Chartier, and Edward F. Chang. “Speech synthesis from neural decoding of spoken sentences”. In: 568.7753 (Apr. 2019), pp. 493–498. DOI: 10.1038/s41586-019-1119-1.

[25] Jonas Kohler et al. “Synthesizing Speech from Intracranial Depth Electrodes using an Encoder-Decoder Framework”. In: *CoRR* abs/2111.01457 (2021). arXiv: 2111.01457. URL: https://arxiv.org/abs/2111.01457.

[26] Jonathan Shen et al. “Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions”. In: *CoRR* abs/1712.05884 (2017). arXiv: 1712.05884. URL: http://arxiv.org/abs/1712.05884.

[27] Miguel Angrick et al. “Speech synthesis from ECoG using densely connected 3D convolutional neural networks”. English. In: *Journal of neural engineering* 16.3 (June 2019). Funding Information: MA, CH, DK and TS acknowledge funding by BMBF (01GQ1602) and NSF (1608140) as part of the NSF/NIH/ BMBF Collaborative Research in Computational Neuroscience Program. MS acknowledges funding by the Doris Duke Charitable Foundation (Clinical Scientist Development Award, grant 2011039), a Northwestern Memorial Foundation Dixon Translational Research Award (including partial funding from NIH National Center for Advancing Translational Sciences, UL1TR000150 and UL1TR001422), NIH grants F32DC015708 and R01NS094748. Publisher Copyright: © 2019 IOP Publishing Ltd. ISSN: 1741-2560. DOI: 10.1088/1741-2552/ab0c59.

[28] Kai Shigemi et al. “Synthesizing Speech from ECoG with a Combination of Transformer-Based Encoder and Neural Vocoder”. In: *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2023, pp. 1–5. DOI: 10.1109/ICASSP49357.2023.10097004.

[29] Xupeng Chen et al. “A Neural Speech Decoding Framework Leveraging Deep Learning and Speech Synthesis”. In: *bioRxiv* (2023). DOI: 10.1101/2023.09.16.558028. eprint: https://www.biorxiv.org/content/early/2023/09/17/2023.09.16.558028.full.pdf. URL: https://www.biorxiv.org/content/early/2023/09/17/2023.09.16.558028.

[30] Ze Liu et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. In: *CoRR* abs/2103.14030 (2021). arXiv: 2103.14030. URL: https://arxiv.org/abs/2103.14030.

[31] Xiaolong Wu et al. “Speech decoding from stereo-electroencephalography (sEEG) signals using advanced deep learning methods”. In: *Journal of Neural Engineering* 21.3 (June 2024), p. 036055. DOI: 10.1088/1741-2552/ad593a. URL: https://dx.doi.org/10.1088/1741-2552/ad593a.

[32] Nicholas S. Card et al. “An accurate and rapidly calibrating speech neuroprosthesis”. In: *medRxiv* (2023). DOI: 10.1101/2023.12.26.23300110. eprint: https://www.medrxiv.org/content/early/2023/12/26/2023.12.26.23300110.full.pdf. URL: https://www.medrxiv.org/content/early/2023/12/26/2023.12.26.23300110.

[33] Pierre Guetschel, Théodore Papadopoulo, and Michael Tangermann. *An embedding for EEG signals learned using a triplet loss*. 2023. arXiv: 2304.06495 [eess.SP]. URL: https://arxiv.org/abs/2304.06495.

[34] Vernon J. Lawhern et al. “EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces”. In: *CoRR* abs/1611.08024 (2016). arXiv: 1611.08024. URL: http://arxiv.org/abs/1611.08024.

[35] Phairot Autthasan et al. “MIN2Net: End-to-End Multi-Task Learning for Subject-Independent Motor Imagery EEG Classification”. In: *IEEE Transactions on Biomedical Engineering* 69.6 (June 2022), pp. 2105–2118. ISSN: 1558-2531. DOI: 10.1109/tbme.2021.3137184. URL: http://dx.doi.org/10.1109/TBME.2021.3137184.

[36] Arjun, Aniket Singh Rajpoot, and Mahesh Raveendranatha Panicker. "Subject independent emotion recognition using EEG signals employing attention driven neural networks". In: *Biomedical Signal Processing and Control* 75 (2022), p. 103547. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2022.103547. URL: https://www.sciencedirect.com/science/article/pii/S1746809422000696.

[37] Xinke Shen et al. "Contrastive Learning of Subject-Invariant EEG Representations for Cross-Subject Emotion Recognition". In: *CoRR* abs/2109.09559 (2021). arXiv: 2109.09559. URL: https://arxiv.org/abs/2109.09559.

[38] Junbo Chen et al. "Subject-Agnostic Transformer-Based Neural Speech Decoding from Surface and Depth Electrode Signals". In: *bioRxiv* (2024). DOI: 10.1101/2024.03.11.584533. eprint: https://www.biorxiv.org/content/early/2024/03/14/2024.03.11.584533.full.pdf. URL: https://www.biorxiv.org/content/early/2024/03/14/2024.03.11.584533.

[39] Rosana Ardila et al. "Common Voice: A Massively-Multilingual Speech Corpus". In: *CoRR* abs/1912.06670 (2019). arXiv: 1912.06670. URL: http://arxiv.org/abs/1912.06670.

[40] Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. "Brain Computer Interfaces, a Review". In: *Sensors* 12.2 (2012), pp. 1211–1279. ISSN: 1424-8220. DOI: 10.3390/s120201211. URL: https://www.mdpi.com/1424-8220/12/2/1211.

[41] Jinhyeok Yang et al. "VocGAN: A high-fidelity real-time vocoder with a hierarchically-nested adversarial network". In: *arXiv preprint arXiv:2007.15256* (2020).

[42] Hendrik Purwins et al. "Deep Learning for Audio Signal Processing". In: *CoRR* abs/1905.00078 (2019). arXiv: 1905.00078. URL: http://arxiv.org/abs/1905.00078.

[43] Paloma Laguarta et al. "Detection of anomalies amongst LIGO's glitch populations with autoencoders". In: *Classical and Quantum Gravity* 41.5 (2024), p. 055004.

[44] Robert F. Kubichek. "Mel-cepstral distance measure for objective speech quality assessment". In: *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing* 1 (1993), 125–128 vol.1. URL: https://api.semanticscholar.org/CorpusID:62230377.

[45] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[46] Xue Ying. "An Overview of Overfitting and its Solutions". In: *Journal of Physics: Conference Series* 1168.2 (Feb. 2019), p. 022022. DOI: 10.1088/1742-6596/1168/2/022022. URL: https://dx.doi.org/10.1088/1742-6596/1168/2/022022.

[47] Alex M Lamb et al. "Professor Forcing: A New Algorithm for Training Recurrent Networks". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper_files/paper/2016/file/16026d60ff9b54410b3435b403afd226-Paper.pdf.

[48] Samy Bengio et al. "Scheduled sampling for sequence prediction with recurrent Neural networks". In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'15. Montreal, Canada: MIT Press, 2015, pp. 1171–1179.

[49] Tsvetomila Mihaylova and André F. T. Martins. "Scheduled Sampling for Transformers". In: *CoRR* abs/1906.07651 (2019). arXiv: 1906.07651. URL: http://arxiv.org/abs/1906.07651.

[50] Cees Taal et al. “A short-time objective intelligibility measure for time-frequency weighted noisy speech”. In: Apr. 2010, pp. 4214–4217. DOI: 10.1109/ICASSP.2010.5495701.

[51] Damien Lejeune and Kurt Driessens. “A data driven similarity measure and example mapping function for general, unlabelled data sets”. In: *Proceedings of the Twenty-Second European Conference on Artificial Intelligence*. ECAI’16. The Hague, The Netherlands: IOS Press, 2016, pp. 158–166. ISBN: 9781614996712. DOI: 10.3233/978-1-61499-672-9-158. URL: https://doi.org/10.3233/978-1-61499-672-9-158.

Appendix A

Appendix

A.1 Speech classification within mel-spectrogram windows

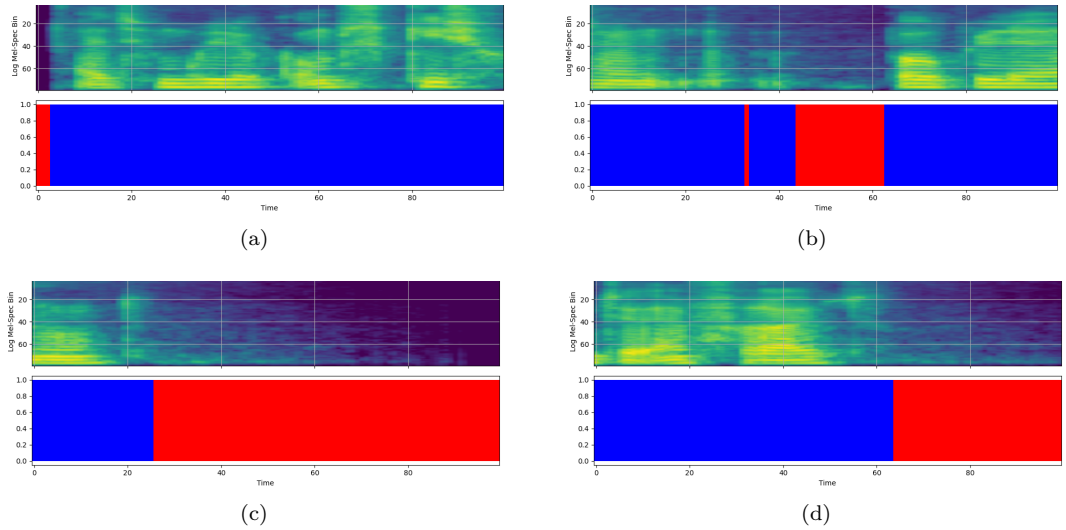
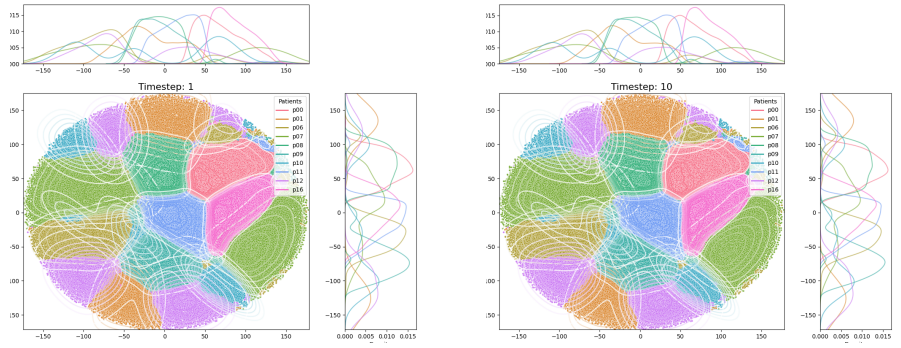


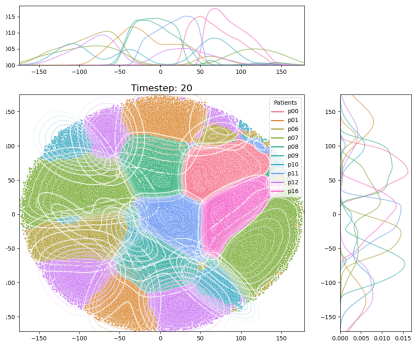
Figure A.1: **Classification of speech within a mel-spectrogram.** The top plot displays the mel-spectrogram, while the bottom plot highlights regions corresponding to speech and silence, with speech segments marked in blue and silence segments in red.

A.2 Autoencoder trajectory lines

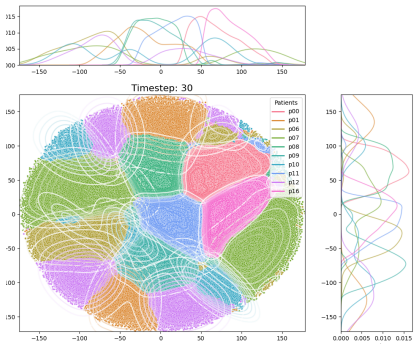


(a) t=0

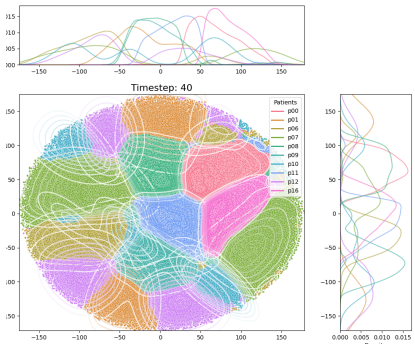
(b) t=10



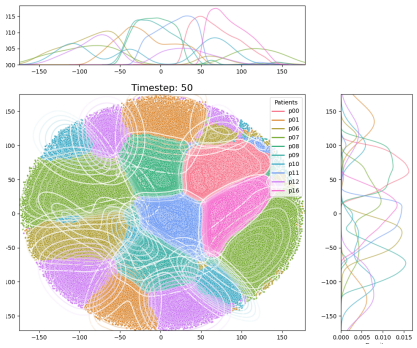
(c) t=20



(d) t=30



(e) t=40



(f) t=50

Figure A.2: Latent data extracted by the autoencoder: t-SNE representation of trajectory lines, illustrating the distribution of testing points across patients at various time points. Each main plot is accompanied by two subplots—one on the top and one on the right—showing the distribution of t-SNE feature 1 and feature 2, respectively. The contour lines found on the main plots provide a visual representation of the density and structure of the underlying point distribution at each specified time point.

A.3 Pearson Correlation

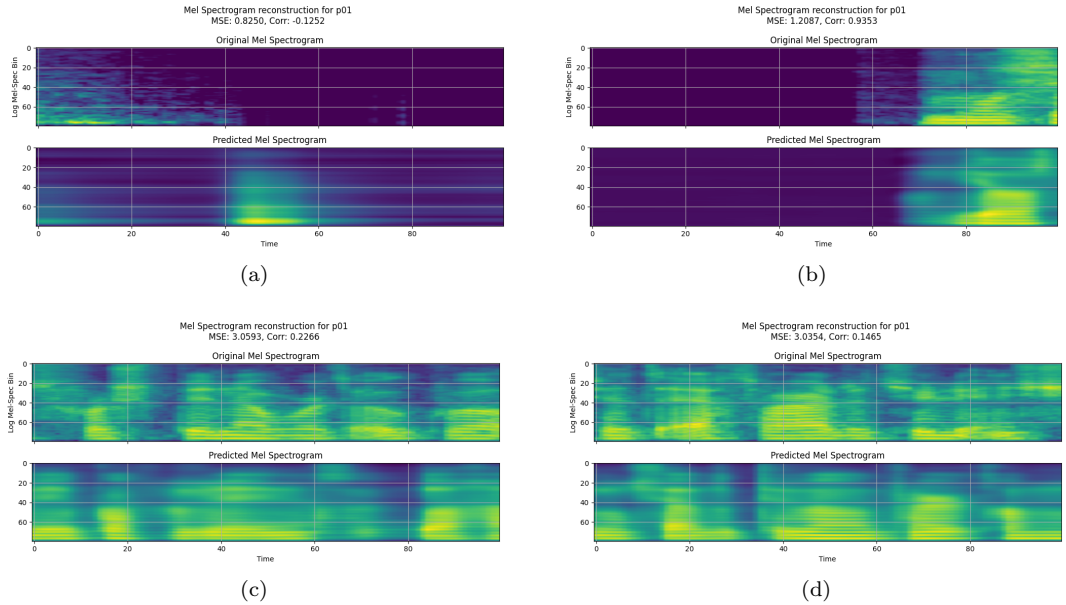


Figure A.3: **Pearson correlation values for mel-spectrograms containing varying amounts of speech within each window.** In each subplot, the top portion shows the original mel-spectrogram, while the bottom portion displays the predicted mel-spectrogram. (a) Mel-spectrogram contains no speech. (b) Mel-spectrogram contains minimal speech. (c) and (d) Mel-spectrograms contain only speech.

A.4 Individual model predictions

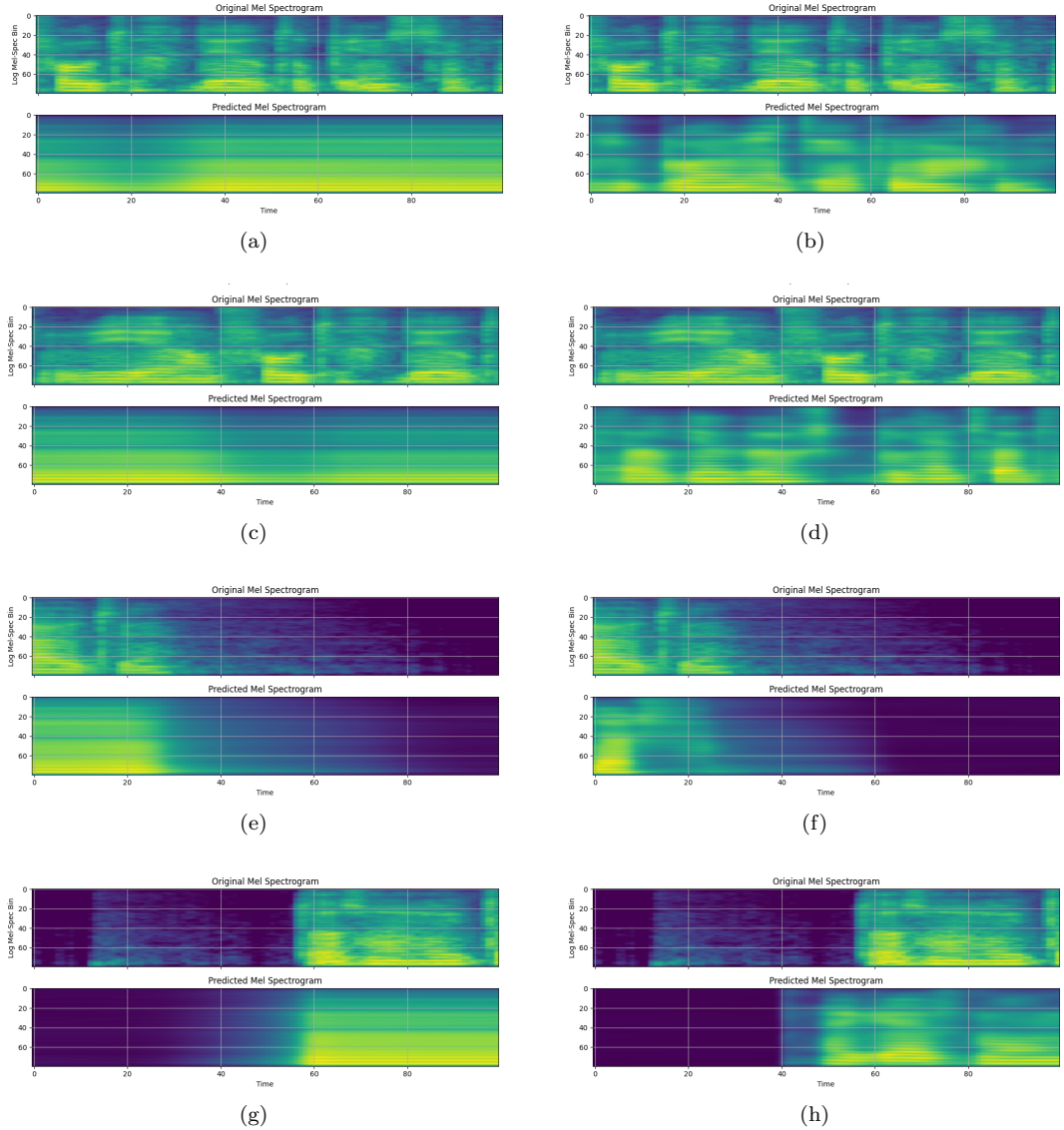
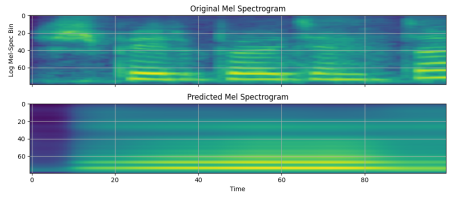
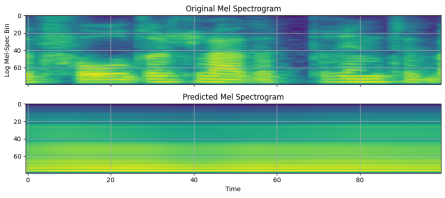


Figure A.4: **Predictions for the normal Individual Model.** The model was trained for 5000 epochs on patient 01 data. Each row of subplots shows predictions for a different test example. The plots on the left display predictions from a model trained for approximately 500 epochs, while the plots on the right display predictions from a model trained for around 5000 epochs.

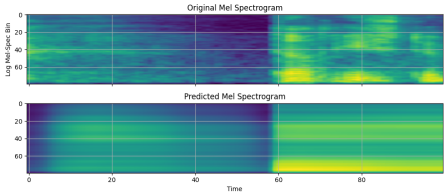
A.5 Cross-patient predictions



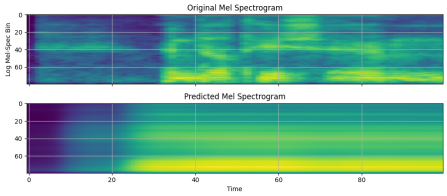
(a) Patient 00



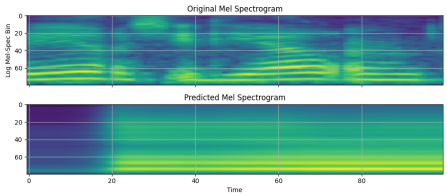
(b) Patient 01



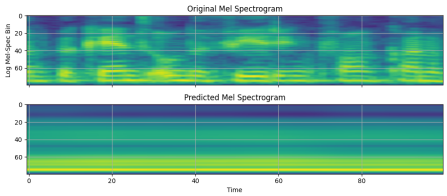
(c) Patient 06



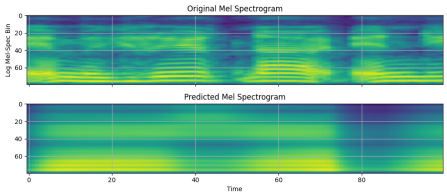
(d) Patient 07



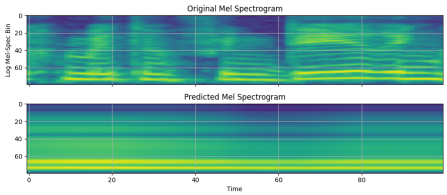
(e) Patient 08



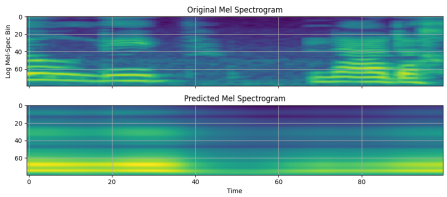
(f) Patient 09



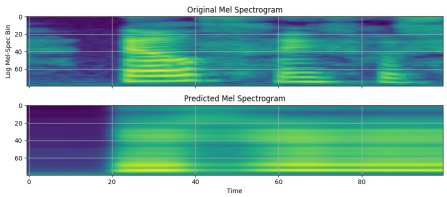
(g) Patient 10



(h) Patient 11

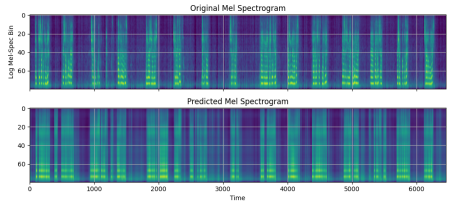


(i) Patient 12

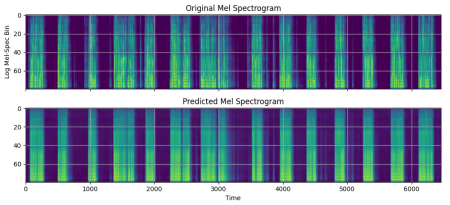


(j) Patient 16

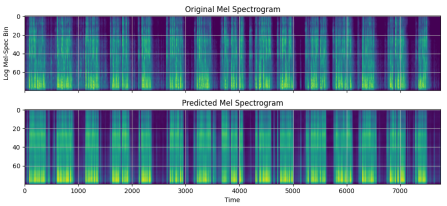
Figure A.5: **Predictions for the normal cross-patient model.** In each subplot, the top portion shows the original mel-spectrogram and the bottom part displays the predicted mel-spectrogram. Each subplot represents predictions for a different patient.



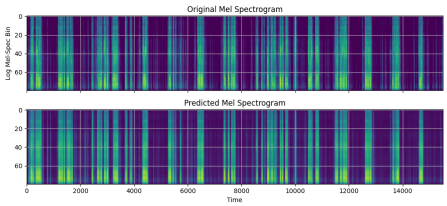
(a) Patient 00



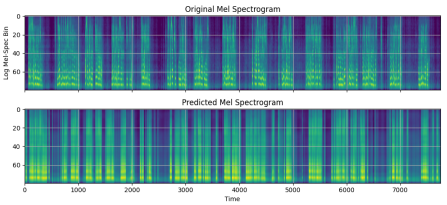
(b) Patient 01



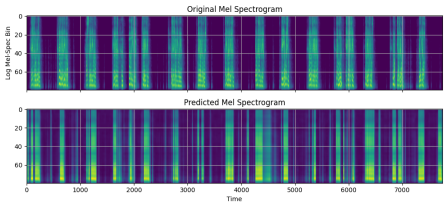
(c) Patient 06



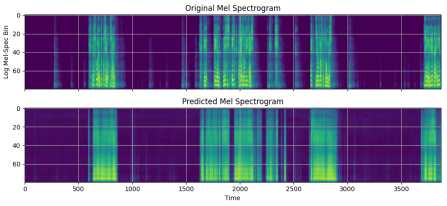
(d) Patient 07



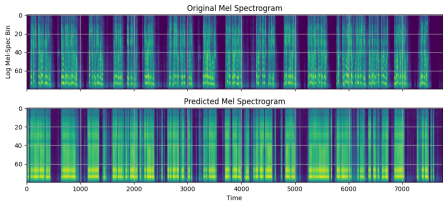
(e) Patient 08



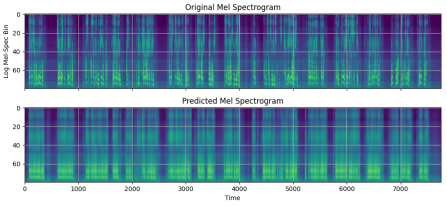
(f) Patient 09



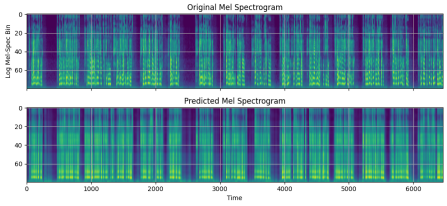
(g) Patient 10



(h) Patient 11



(i) Patient 12



(j) Patient 16

Figure A.6: **Predictions for the normal cross-patient model over the entire test set.** In each subplot, the top portion shows the original mel-spectrogram and the bottom part displays the predicted mel-spectrogram. Each subplot represents predictions for a different patient.

A.6 Pre-trained model predictions

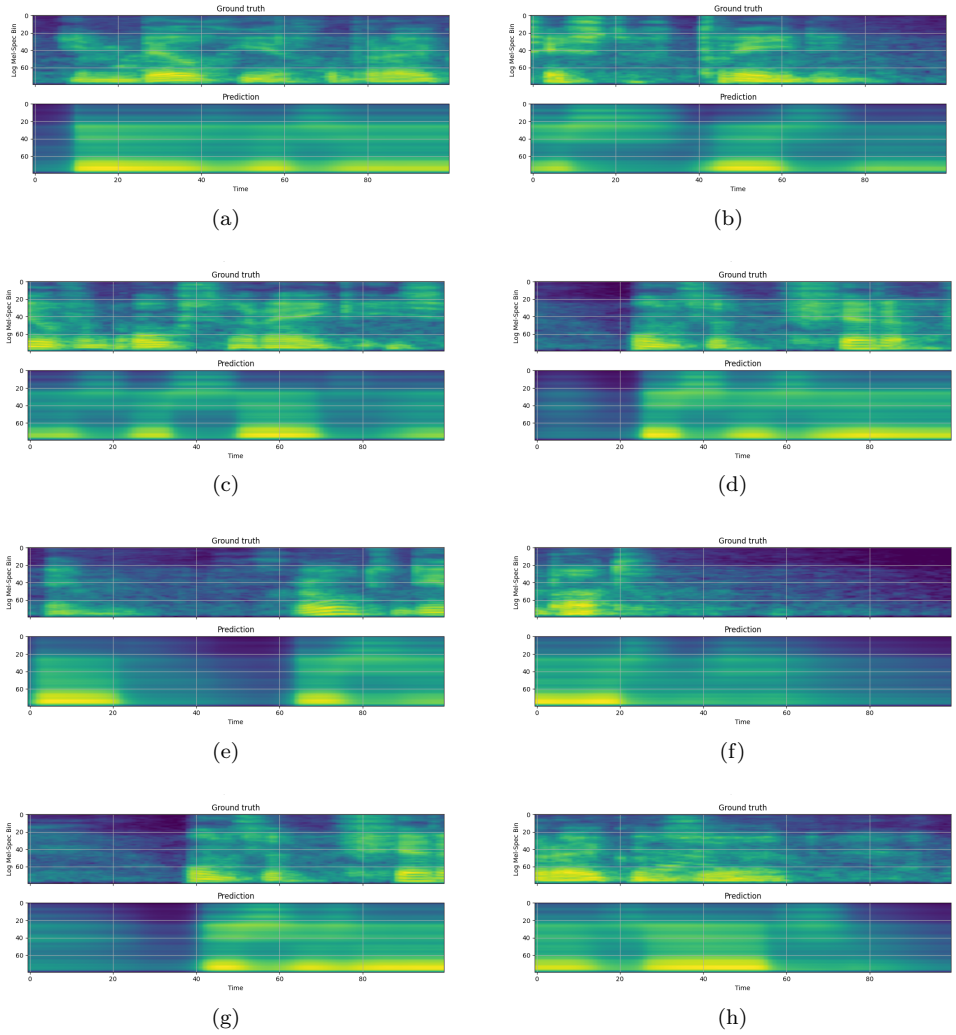


Figure A.7: **Predictions for the Normal Pre-trained Model.** The decoder in this model was initially trained within an individual model on patient 01 data for 500 epochs after which it trained for an additional 500 epochs on patient 06 data. In each subplot, the top part displays the original mel-spectrogram, while the bottom part shows the predicted mel-spectrogram. Each subplot represents predictions for a different patient.

A.7 Overall results on stitched mel-spectrograms

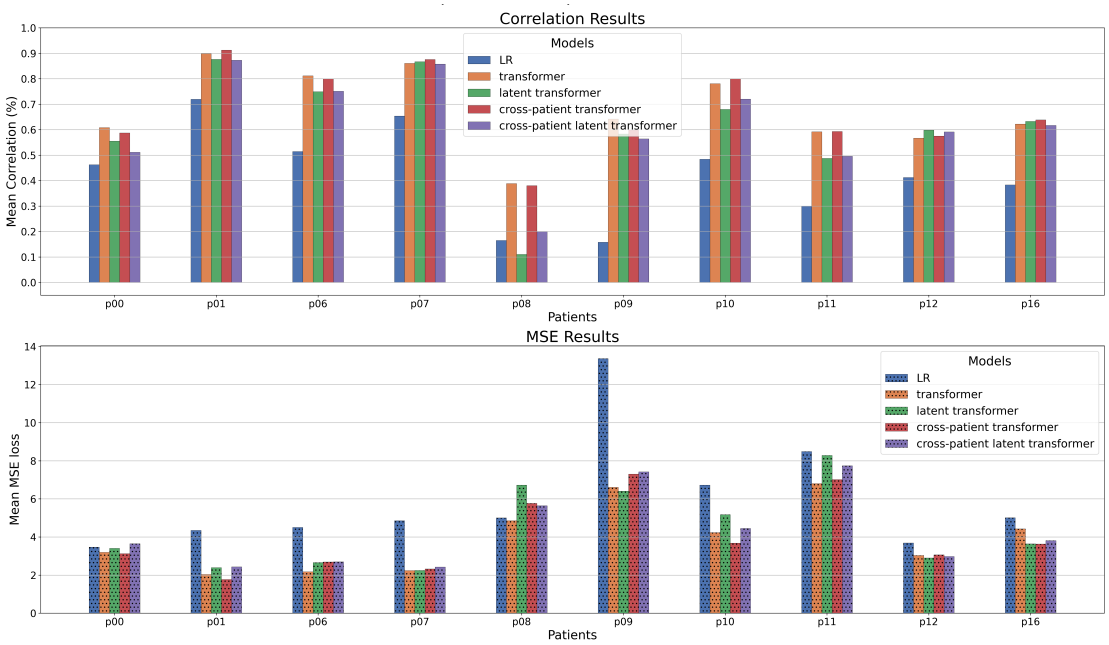


Figure A.8: Comparison of model performance across different datasets using various models: **LR**, **transformer**, **latent transformer**, **cross-patient**, **latent cross-patient**, and **latent transformer**. To obtain these results, each model generated predictions on windowed mel-spectrogram segments, which were then stitched together to form one, complete test mel-spectrogram. Correlation and MSE metrics were calculated using the resulting stitched spectrogram. The top plot shows the correlation results, while the bottom plot displays the MSE results.