Bachelor thesis

Artificial Intelligence

Radboud University

Biologically Plausible Neural Network Models of Prelexical Speech Processing

Author:
Floris van Wettum
s1008742

First supervisor:
Dr. H. Fitz
Donders Institute for Brain, Cognition and Behaviour
Hartmut.Fitz@mpi.nl

Second supervisor:
PhD candidate A. Quaresima
Max Planck Institute, Neurobiology of Language
Alessio.Quaresima@mpi.nl

June 18, 2021

Abstract

Speech recognition requires mapping highly variable sound waves into concrete units of speech. By means of this mapping, the correct words can be reconstructed. However, which exact speech units are learned for prelexical processing has been an area of debate. To investigate this matter, we simulate a low-level area in the auditory pathway of the brain, by means of a spiking neural network (SNN), that receives its input from the cochlea. This SNN learns through the unsupervised spike-timing-plasticity learning paradigm (STDP). It is hypothesised that the representations learned by the SNN are best compared to phones because the network receives low-level input. The responses of SNNs to specific stimuli are analysed in order to investigate the representations learned by these types of networks. Responses of SNNs are compared by employing a custom-defined 'response discrepancy' measure. This measure computes the mean distance between the average responses, of for example different SNNs, to the same class of stimuli, thus quantifying their dissimilarity. The response discrepancy measure is used to compare the responses of a single SNN to two different test sets, as well as to compare the responses of two differently trained SNNs to the same test set. This analysis of the response dissimilarities showed no generalisation of the network's responses to phones or words uttered by different speakers from different dialects or genders. This suggests that the hypothesised unit of phones is not learned by the SNN after learning.

Acknowledgements

First and foremost, I would like to thank my supervisors Hartmut Fitz and Alessio Quaresima for introducing the topic of the project and their guidance throughout the entire process. A special thanks to Alessio for joining efforts with us on writing a `Julia` module that allowed us to interact with the dataset and for helping on the more pratical side of the project.

Next, I would like to thank my group members: Joost Kasper, Fleur Brandsen and Léonie Wagner with whom I did a large part of the project, which includes the pre-processing of the data, extending the SNN model, the classifier, and everything that is involved in making these parts work.

Finally, I would like to thank the people who proof-read my work. Thanks to my brother Yannick van Wettum, sister Martine van Wettum, girlfriend Christine Li, and also my supervisors Hartmut Fitz and Alessio Quaresima for the feedback provided on my written work and great overall support.

Contents

1. Introduction

1.1 Problem statement and background

Verbal communication is a fundamental component of human interaction and information exchange. Countless times per day we hear, and process spoken words. To a large extent, this is done automatically and in a highly efficient manner. The task of recognising what has been said is known as 'speech recognition'. Speech recognition requires solving the invariance problem (Pisoni, 1985). The invariance problem refers to the problem of mapping enormously variable speech signals (as a function of dialects, style differences, etc.) into concrete units of speech, as intended by the speaker (Myers et al., 2017). The central question then is, what are these units of speech that are learned by the brain? The exact representations that are learned by the auditory cortex (AC) for speech recognition is still a matter of debate (Mitterer et al., 2018). Some argue that allophones are the building blocks that the brain uses for spoken word recognition (Mitterer et al., 2018) while others argue for phonemic units (Bowers et al., 2016).

To give an overview of the aforementioned and other necessary phonology terms, *phonemes* are abstract mental representations of speech sounds which are never physically produced and distinguish words with different meanings (Yi et al., 2019). Likewise, *allophones* are phonetic variations of phonemes, they describe different sounds within this phoneme (Mitterer et al., 2018). Naturally, there is the physical realisation of the sound, these sound units are referred to as *phones*. Lastly, there is the notion of *acoustic-phonetic* features, also called *sub-phonetic* features, that describe the underlying acoustic properties and articulatory gestures that generate these speech sounds (see Appendix Section A.1 for examples) (Yi et al., 2019). Because speech processing is done in a hierarchical structure, these different units of speech (e.g., phones, allophones, phonemes) may be relevant at different stages of processing (Yi et al., 2019). Therefore, evidence for the use of one unit does not rule out the use of another unit elsewhere (Mitterer et al., 2018).

At the beginning of the speech processing pipeline, there is the *cochlea* in the inner ear which converts sound waves into time-frequency representations, encoded by neuronal spikes (Yi et al., 2019). These neuronal spikes are a result of sensory neurons in the cochlea being sensitive to specific frequencies (Pan et al., 2020). This means that the firing of specific sensory neurons is indicative of the presence of specific frequency bands in the perceived sound signal (Liberman, 1982). The neural activity of these frequency sensitive neurons is the input of the prelexical processing stage, where the low-level neural speech signals are converted into concrete units of speech.

1.2 State of the field

Speech recognition algorithms facilitate controlling devices by means of speech. A great example of this is the recent rise in smart home appliances and virtual assistants. The most remarkable Artificial Intelligence (AI) algorithms for speech recognition created today are achieved through neural networks or deep neural networks. These Artificial Neural Networks (ANNs) have proven to be capable of achieving superhuman performance for narrow tasks (Aston Zhang & Smola, 2021). Despite taking inspiration from the brain, ANNs are not biologically plausible as they rely on real valued activations to be passed and use non-local information for learning (Dong et al., 2018). The next step in AI and speech recognition could be to move towards more biologically plausible models. Such models are interesting

because their mechanisms are comparable to that of the brain. This allows them to be used as bottom-up approaches for understanding the brain through experimentation and analysis. In addition, it is possible to implement these biologically plausible methods in energy efficient specialised hardware, an example of this is the TrueNorth neurosynaptic chip (Akopyan et al., 2015). Furthermore, the fact that audio is a temporal signal, just like spike trains (Section A.3), makes Spiking Neural Networks (SNNs) a natural choice for dealing with dynamic signals such as audio. Litwin-Kumar & Doiron (2014) have shown that such biologically realistic SNNs can learn stable memories of before seen stimuli. This SNN learns by means of an unsupervised spike-timing-dependent plasticity (STDP) learning paradigm (Clopath et al., 2010), which is also found in the brain (Bi & Poo, 1998). This learning paradigm is a realisation of the Hebbian learning rule (Clopath et al., 2010), which is often summarised as neurons that "fire together, wire together, and those that fire out of sync, lose their link". In other words, if the firing of one neuron appears to be causal for the firing of another neuron, they will form stronger connections. However, if their firing times are largely uncorrelated, their connection will weaken.

1.3 Research

In this thesis, we will extend the SNN model proposed by Litwin-Kumar & Doiron (2014) to take in temporal speech signals that have been pre-processed in a biologically plausible manner (Pan et al., 2020). This SNN then simulates a low-level area in the auditory pathway that receives low-level input directly from the cochlea.

The dataset used for this is the Spike-TIMIT dataset as proposed by Pan et al. (2020). This dataset extends the original TIMIT dataset (Garofolo, 1993), converting its sound files into neuronal spikes encoded by 620 neurons. The TIMIT dataset contains many speakers from different American English dialect regions uttering phonetically rich sentences (Garofolo, 1993). As speakers from different dialect regions or genders differ in pronunciation, it is a great tool to be able to selectively present subsets of speakers to the model. Nevertheless, the most important aspect of the Spike-TIMIT dataset is that it is pre-processed in a biologically plausible way. This pre-processing follows the three criteria for Biologically plausible Auditory Encoding (BAE), emulating the functions of the perceptual components of the human auditory system (Pan et al., 2020). Given that the data is pre-processed in a biologically plausible way and the area that receives this data is simulated by a biologically plausible model, more reliable inferences can be made about the brain. From this dataset, specific words and the phones that constitute them are extracted and fed into the network. The network's responses therefore carry the label of the word or phone to which this response was measured.

By feeding the responses of the network to stimuli into a classifier, we simulate a speech recognition pipeline. However, for the classifier, no biological plausibility is considered in this research and it mainly functions as a means of probing the network. I will then individually analyse the activity of these SNNs, trained on different dialects or genders, to answer the central research question.

1.3.1 Aim and hypothesis

The aim of this thesis is to learn about SNNs and to study what representations are learned by the SNN described above. This would also show whether the representations learned by the SNN are similar to the representations one might find in the simulated area in the human brain. Since we use an SNN to simulate an area that receives low-level input, the hypothesis is that the speech units learned by the network are not highly abstract and are best compared to phones or allophones rather than phonemes or words.

1.4 Outline

In Section 2, I will introduce the methods used for data pre-processing and give an overview of the network and analysis done on its activity. In Section 3, I will present the results gathered using these methods. Next, I will compare this work to related work and make suggestions for future research in Section 4. Lastly, the conclusions drawn from this research are listed under Section 5.

2. Methods

This section describes how the data is pre-processed and used, what the architecture of the SNN is, and how the activation of SNNs can be analysed to compare responses of these SNNs.

2.1 Pipeline overview

A rough overview of the entire pipeline is given in Figure 2.1. Here, the stages of the model evaluations are depicted. First, a newly initialised network is trained on a (sub)set of data. After learning, as the weights have started to stabilise, the weights are saved. This is where learning of the SNN is turned off and test data is fed to the network. The activity of the network, its membrane potentials, are measured and saved for each stimulus presentation. Once the test set has been exhausted, the measured network states are once again split in a train and test set for the classifier to train and test on. Lastly, the set of all measured states is used to analyse the activity of the model, which will be discussed in Subsection 2.4.

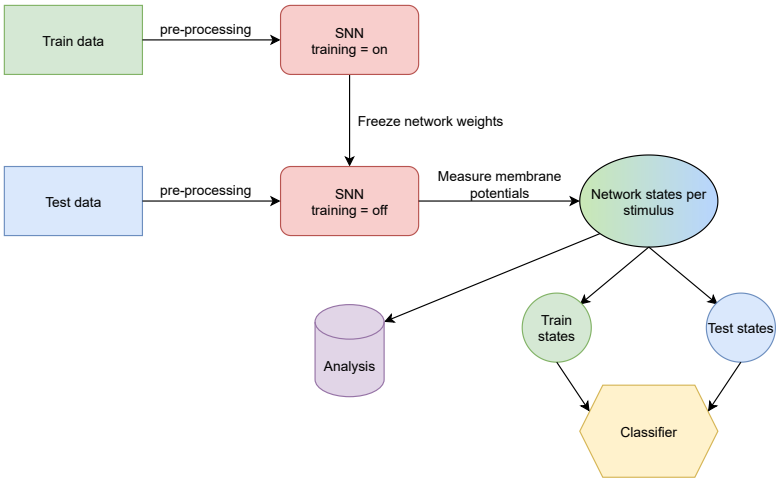


Figure 2.1: Overview of the training and testing pipeline. The figure shows the different components of the project. The colours bear no specific meaning and are used for visualisation purposes.

2.2 Data encoding

The original data from the Spike-TIMIT dataset is encoded by spike trains in 620 neurons, following a sparse coding scheme and thus encoding information with few spikes. These neurons simulate sensory neurons in the cochlea (Pan et al., 2020) and can be divided into groups of 31 neurons. Each group of 31 neurons encodes a single frequency band and each individual neuron encodes for a specific spectral energy threshold within this frequency band (Pan et al., 2020). This type of encoding is known as threshold coding (Gütig & Sompolinsky, 2009). Originally, each of these 620 neurons projected to a different group of neurons within the SNN. The group of neurons within the SNN that receive the same input is called a population, or assembly. However, the used sparse coding scheme proved to not evoke enough neuronal activity for the STDP learning to alter the weights under the current parameterisation of the learning paradigm (see Section 3.1). Here two extra pre-processing steps are introduced to battle this lack of evoked potential.

2.2.1 Population grouping

For the first pre-processing step, the spike trains of multiple data encoding neurons within the same group (encoding for a single frequency band) are combined into a single spike train. Conceptually this means that the original 620 populations, which each received input from a different data encoding neuron, are combined into larger new populations. In this new population, all neurons receive all spikes that would originally only go to a subset of neurons within this population, thus increasing the evoked potential. The intuition behind this is depicted in Figure 2.2. The exact implementation of this pre-processing step is given in Appendix Section A.6.

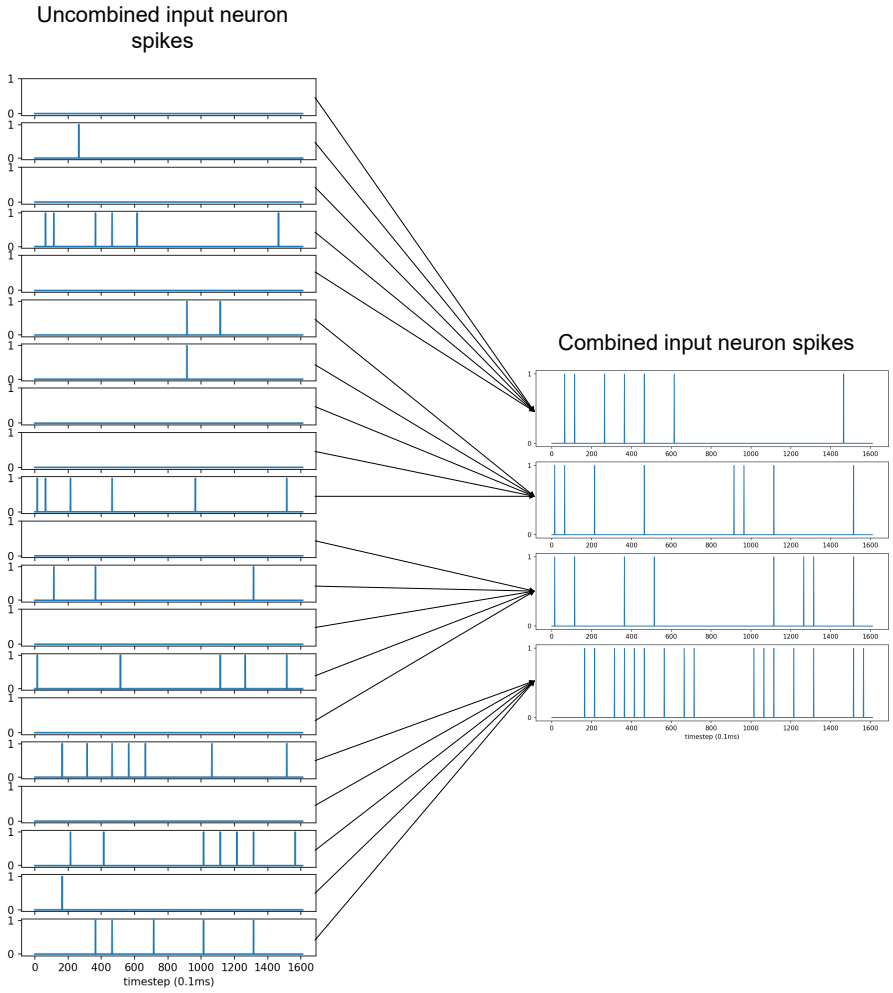
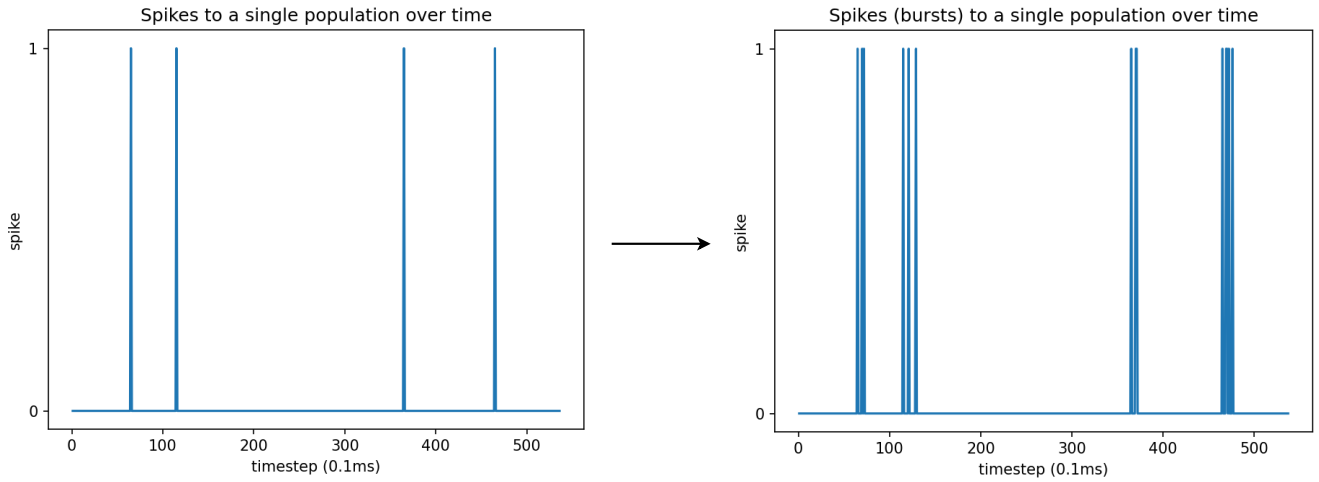


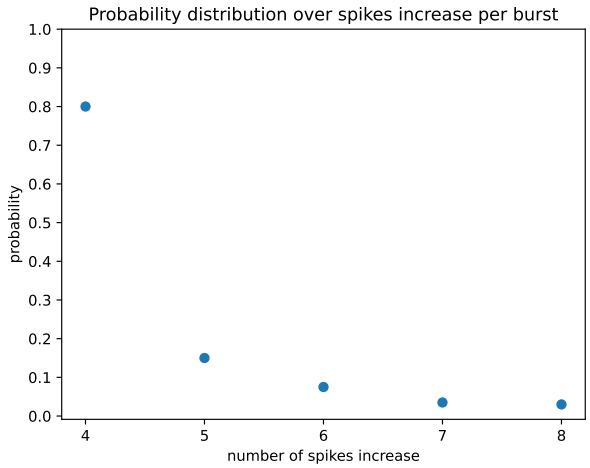
Figure 2.2: Schematic visualisation of combining multiple input spike trains of different data encoding neurons into new spike trains that will be projected into the combined population.

2.2.2 Burst encoding

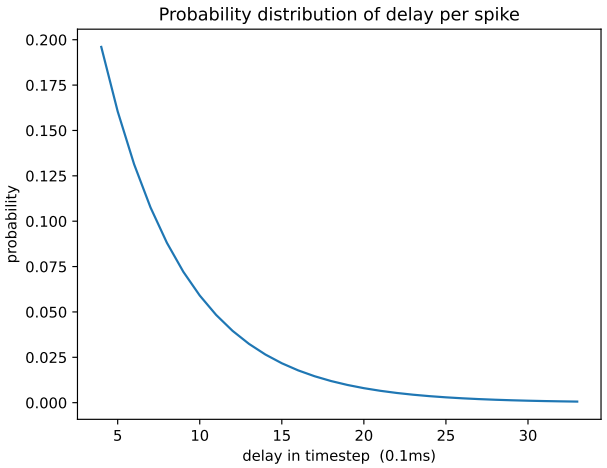
The second pre-processing step involves converting the spikes in these newly obtained spike trains into bursts (see Figure 2.3a). These bursts start at the time of the original spike and have a variable number of spikes that constitute them, as shown in Figure 2.3b (Oswald et al., 2007). The bursts preserve the temporal structure of the data and are also found in different parts of the brain (Zeldenrust et al., 2018).



(a) Visualisation of the conversion of single spikes into bursts.



(b) Probability distribution of the number of spikes added by the burst.



(c) Probability distribution of the delay of each added spike after the previous spike.

Figure 2.3: **Conversion of single spikes into bursts.** (a) An example of what the spike train looks like after conversion of spikes into bursts. (b) The probability distribution of how many spikes are added to the original spike as a result of the burst encoding. (c) The probability distribution of delay with which each individual spike is added to the burst.

2.3 Network

2.3.1 Network and neuron model

The SNN used in this research consists of 5000 neurons, 4000 of which are excitatory adaptive exponential integrate-and-fire neurons (Brette & Gerstner, 2005), and 1000 are inhibitory integrate-and-fire neurons (Litwin-Kumar & Doiron, 2014). These neurons are connected to every other neuron in the network with a probability of 0.2. Each of these connections is initialised with a weight depending on the type of the presynaptic and postsynaptic neuron that form this connection, also called the connection type. The table of initial values per connection type is given in Appendix A.2.1. For a new network, a number of neuron populations is created. The number of populations is decided by the number of different input spike trains. There are as many populations as spike trains created by the pre-processing step described in Section 2.2.1. Each excitatory neuron in the SNN belongs to any specific population with a probability of 0.1. This means that neurons can be members of multiple

populations. Since these populations are only filled with excitatory neurons, only excitatory neurons get targeted by the input data.

To keep the network in an excitable state, background activity is fed into the network. This background activity entails a constant rate of spikes into each neuron of the SNN. For the excitatory neurons, this rate is 4.5kHz with a synaptic weight of 1.78pF while the inhibitory neurons receive 2kHz of background activity spikes with a synaptic weight of 1.27pF. The original model as described by Litwin-Kumar & Doiron (2014), makes use of a population coding scheme, where the input data was encoded with an 8kHz signal (Litwin-Kumar & Doiron, 2014). Population coding entails that information of stimuli are encoded by what population of neurons is activated (Wu et al., 2002). However, in this research, a sparse temporal population coding scheme is used. This means that information is also encoded in the relative timing of the spikes (Thorpe, 1990). Due to the sparsity of the spikes in the data under this coding scheme, compared to the relatively high background activity spike rates, the synaptic weight with which the data signal is fed into the SNN is set to 100pF, instead of the original 1.78pF for excitatory neurons. All aforementioned values can also be found in Appendix Section A.2.1.

Each neuron in the network has its own membrane potential v, these membrane potential dynamics are given by the functions described in Appendix Section A.4. When the membrane potential of a neuron exceeds the threshold, it elicits a spike, as depicted in Figure 2.4. The neuronal spikes of all the neurons in the network over a short interval of time is shown in Figure 2.5. Furthermore, each neuron also has an adaptation current w_{adapt} which functions as a form of local neural memory, also described in Appendix Section A.4.

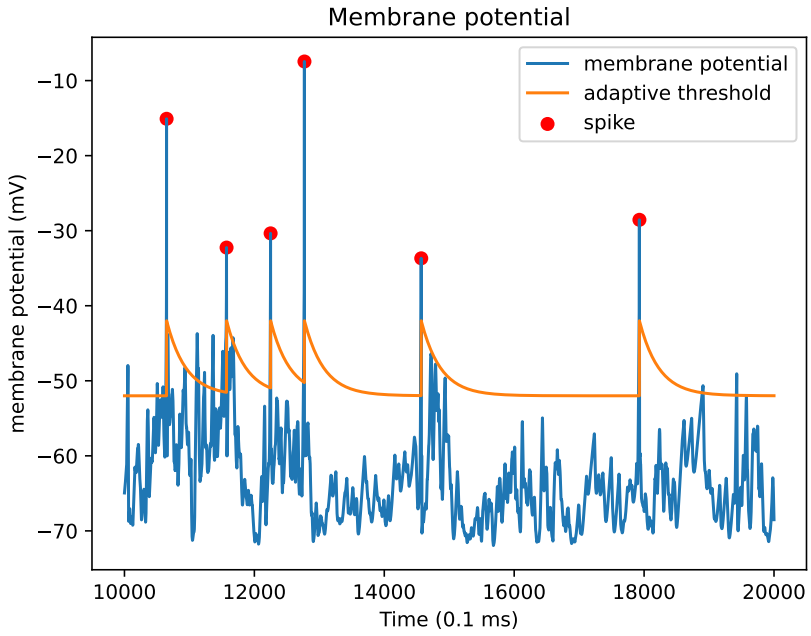


Figure 2.4: Membrane potential, adaptive threshold and the emitted spikes of a single excitatory neuron over an arbitrary time interval.

2.3.2 Learning

The network's synaptic weights are modified by the Hebbian STDP learning paradigm (Clopath et al., 2010). Hebbian STDP is applied to excitatory-excitatory connection weights. This STDP learning paradigm is extended with inhibitory spike-timing-dependent plasticity (iSTDP) (Vogels et al., 2011), which functions as a homeostatic mechanism that offers stability to the learned neural assemblies (Vogels et al., 2011; Litwin-Kumar & Doiron, 2014). This iSTDP rule governs the weights of inhibitory-excitatory connections. The weights of excitatory-inhibitory and inhibitory-inhibitory connections are not plastic. Even though STDP alters the synaptic weights of excitatory-excitatory connections

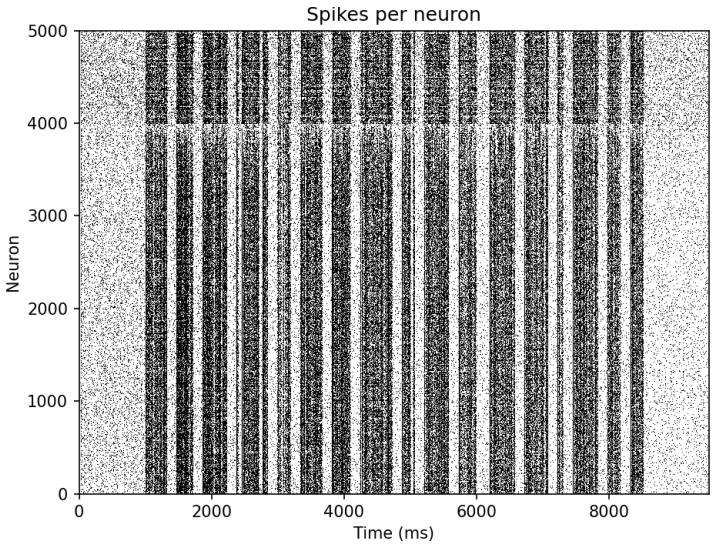


Figure 2.5: Spikes of all neurons in the network over a short interval of time. The neurons are on the vertical axis, whereas the horizontal axis denotes time in ms. Each black dot denotes a spike emitted by a neuron. Here 17 words were presented. Each vertical bar of densely packed dots is the response of the network to presentation of a word, evoking more potential and causing neurons to fire more often.

within certain bounds, these excitatory connection weights also undergo 'synaptic normalisation'. This synaptic normalisation entails that all synaptic weights entering a single neuron should sum to a constant (Litwin-Kumar & Doiron, 2014). All formulae and parameters governing the learning of the network are described in Appendix Section A.2.

2.4 Analysis

2.4.1 Measurements

Both the membrane potential and the adaptation current (v, w_{adapt}) of each neuron is measured 10 times during the presentation of each word and phone. These measurements are uniformly distributed over the duration of the stimulus and thus result in a feature vector of 80000 features ($10 * 4000$ v values and $10 * 4000$ w_{adapt} values). To limit the feature space for further analysis, these values are averaged per neuron to get an average membrane potential $\bar{v}$, and adaptation current $\bar{w}_{\text{adapt}}$ per neuron over a stimulus presentation. This averaging reduces the dimensionality of this feature vector to 8000 dimensions and thus contains the average v and w_{adapt} values for each neuron over the stimulus presentation. These reduced feature vectors are used in subsequent analyses and are referred to as responses, or projections, of the SNN to a stimulus.

2.4.2 Classification

To quantify the separability of data points (responses) of different classes, the mean classification score of a multinomial logistic regression classifier is used. Data points that belong to the same class are responses of the network to stimulus presentations of the same class label, so the same word or phone. This mean classification score is obtained by averaging over the classification scores after training and testing the classifier anew ten times on newly sampled train and test sets, taken from the measurements (states) as described in Section 2.1. To add to this, confidence intervals of 95% of these mean classification scores are computed by means of the formulae stated in Appendix Section A.7.

2.4.3 Network response comparison

The difference in responses of SNNs can be quantified by means of a response discrepancy measure. This measure is based on distance measures to quantify the dissimilarity of the responses of SNNs to the same or similar stimuli. Manhattan distance is used as the distance metric, which is shown to be

preferable over Euclidean distance when working with high dimensionality (Aggarwal et al., 2001).

When computing the response discrepancy between two SNNs given the same data, the Manhattan distance from the response of one network to the response of the other network corresponding to the same stimulus could be computed. This method is described in Formula 2.1. However, when comparing the responses of a single SNN to two different sets of data, for example two different dialects, the responses to stimulus presentations cannot be directly compared as each utterance is only in one of the data sets. Therefore, a more general version of the response discrepancy measure is introduced as described in Formula 2.2, which compares class means instead. The response discrepancy measure computes the mean distance between the average responses, taken from the two different projections (collection of responses), to all stimuli of a class. This more general version is used to compute **all** response discrepancies mentioned in this thesis to more reliably compare response discrepancies across conditions.

$$\text{response discrepancy}(\mathbf{R_A}, \mathbf{R_B}) = \frac{1}{C} \sum_{c=1}^{C} \sum_{s=1}^{S_c} \sum_{i=1}^{d} |\boldsymbol{r}_i^s - \boldsymbol{r}_i'^s| \tag{2.1}$$

$$\text{response discrepancy}(\mathbf{R_A}, \mathbf{R_B}) = \frac{1}{C} \sum_{c=1}^{C} \sum_{i=1}^{d} |\boldsymbol{\mu}_i^c - \boldsymbol{\mu}_i'^c| \tag{2.2}$$

Where $\mathbf{R_A}$ and $\mathbf{R_B}$ are the two collections of responses of SNNs to stimuli. C is the number of classes, S_c is the number of stimuli of class c, d is the dimensionality of the feature vectors, $\boldsymbol{r}^s$ is the feature vector of the response to stimulus s, taken from $\mathbf{R_A}$, $\boldsymbol{r}'^s$ is the feature vector of the response to the exact same stimulus s, taken from $\mathbf{R_B}$, $\boldsymbol{\mu}^c$ is the mean feature vector of the responses to stimulus class c, taken from $\mathbf{R_A}$ and $\boldsymbol{\mu}'^c$ is the mean feature vector of the responses to stimulus class c, taken from $\mathbf{R_B}$.

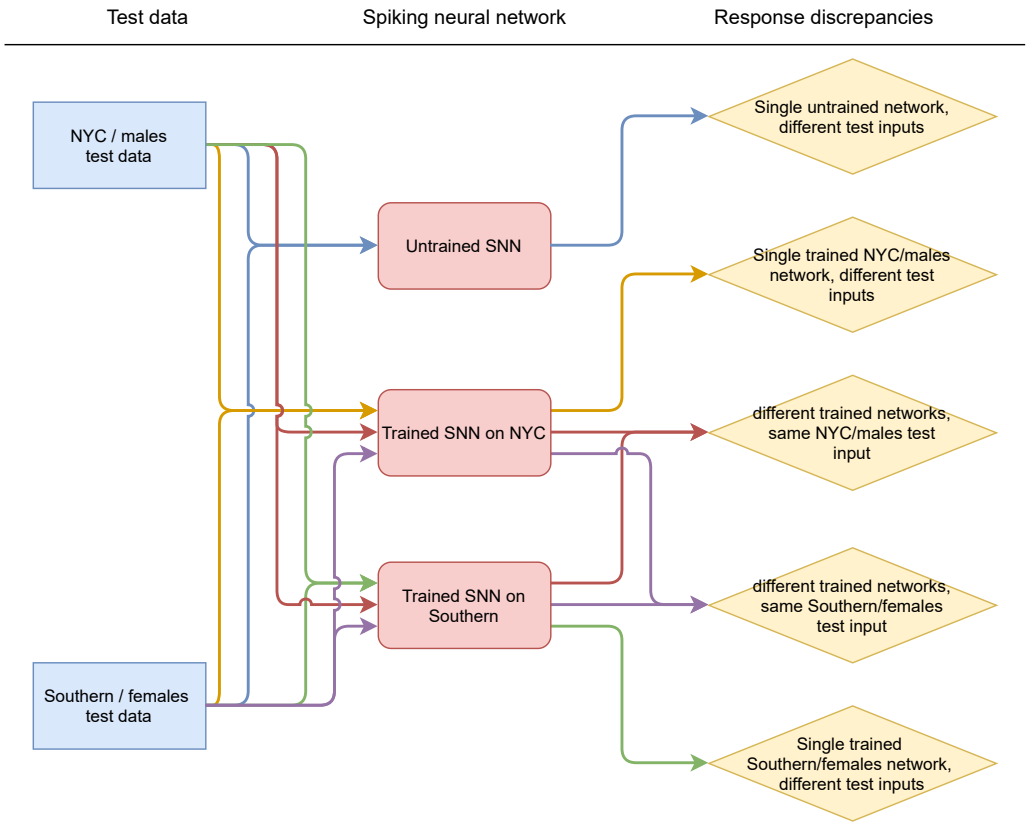


Figure 2.6: Overview of the tests that have been conducted to compute specific response discrepancies.

To investigate the precision and generalisation properties of the network on the units of phones and words, five response discrepancies can be computed. These response discrepancies describe how

dissimilar the responses are of the same untrained SNN given two different test sets (different dialects or genders), the same trained SNN given two different test sets (different dialects or genders), an untrained and trained SNN given the same test set (a single dialect or gender), two trained SNNs trained on different training sets (different dialects or gender), given the same test input. This is better summarised in Figure 2.6. This experiment is further divided into more specific comparisons, following the testing schemes depicted in Figures 2.7 and 2.8. These testing schemes are to compare the responses of a single network to different test sets, and the responses of different networks on the same test set respectively.

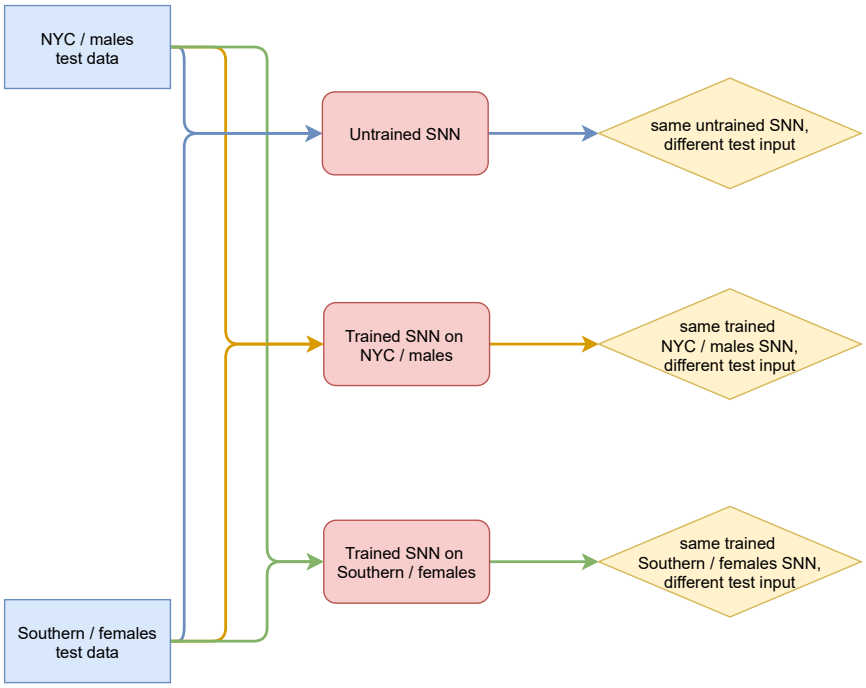


Figure 2.7: Overview of the tests that have been conducted to compute response discrepancies between the responses of a single SNN on two different test sets.

Another way to quantify the similarity of networks is by comparing either the weight distributions of these networks or the the eigenvalue distributions of the responses of these networks given the same input. To compare these distributions, the Hellinger distance (Nikulin, 1994) and a symmetric Kullback-Leibler divergence, as described in Formula 2.3, are computed.

$$\frac{1}{2}(D_{KL}(P||Q) + D_{KL}(Q||P)) \tag{2.3}$$

2.4.4 Separability of responses

The effect that learning has on the separability of the responses to different stimulus classes is also analysed. To look at the separability, the within and between class mean absolute deviations (MADs) (Taleb, 1970) in the high dimensional feature space can be of interest. The within class MADs are defined as the mean distance from all data points (responses) that belong to the same class, to the mean of that class. For example, the within class MAD for the word "water" is the mean distance from all response feature vectors when presented with the word "water" to the average response to that word. By averaging over the within class MADs of each class, a mean within class MAD is computed, as described in formula 2.4. The between class MADs are given by the mean distance from one class mean (mean response to a stimulus class) to all other class means, for all classes. Again, by averaging over all the between class MADs of each class, the average between class MAD is obtained, as described in Formula 2.5.

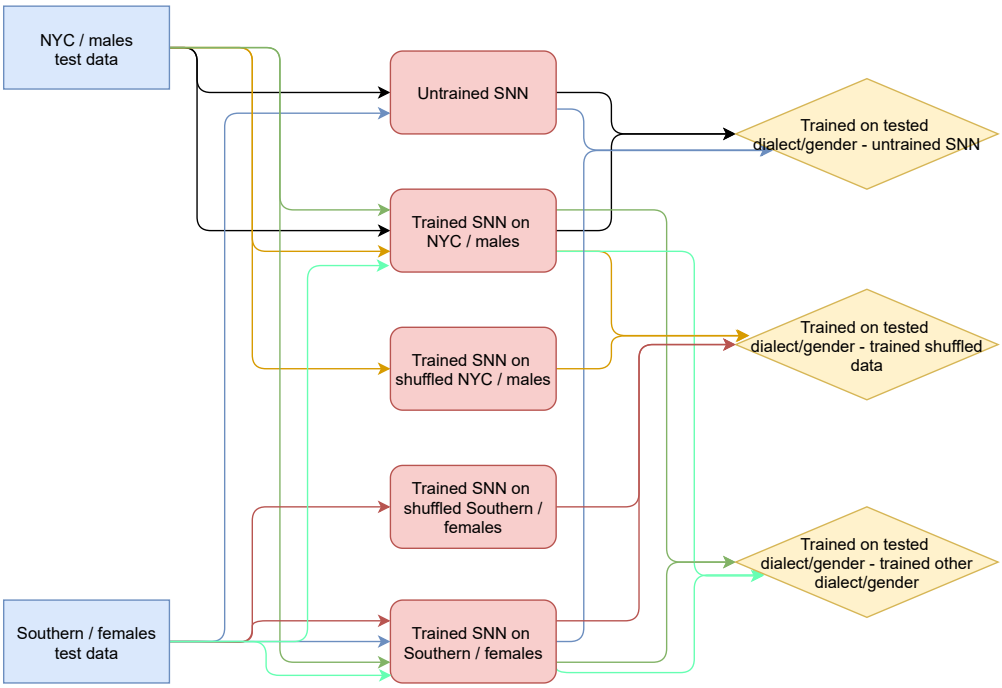


Figure 2.8: Overview of the tests that have been conducted to compute response discrepancies between the responses of two different SNNs using the same test set.

$$\text{mean within class MAD} = \frac{1}{C}\sum_{c=1}^{C}\sum_{s=1}^{S_c}\sum_{i=1}^{d}|\boldsymbol{r}_i^s - \boldsymbol{\mu}_i^c| \tag{2.4}$$

$$\text{mean between class MAD} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{C-1}\sum_{k=1}^{C}\sum_{i=1}^{d}|\boldsymbol{\mu}_i^c - \boldsymbol{\mu}_i^k| \tag{2.5}$$

Where C is the number of classes, S_c is the number of stimuli of class c, d is the dimensionality of the feature vector, $\boldsymbol{r}_i^s$ vector which represents the response of the network to stimulus s and $\boldsymbol{\mu}^c$ is the mean feature vector of class c.

2.4.5 Effective dimensionality

Lastly, to compare the shape of the data in high dimensional feature space of different responses, the $n1$ index Effective Dimensionality (ED) (Del Giudice, 2020) is used. This measure is also described in Formula 2.6. The effective dimensionality describes the equivalent numbers of orthogonal dimensions that would produce the same overall pattern of covariation (Del Giudice, 2020). In other words, how effectively does the data use the dimensionality of the space? If the data is projected onto a flat disk in three-dimensional space, the ED will be below three as it does not use this third dimension as effectively as the others, which contain more variance. Whereas, if the data is projected onto a shape that has uniform variance in all directions, the ED will be equal to the number of dimensions of the feature vector.

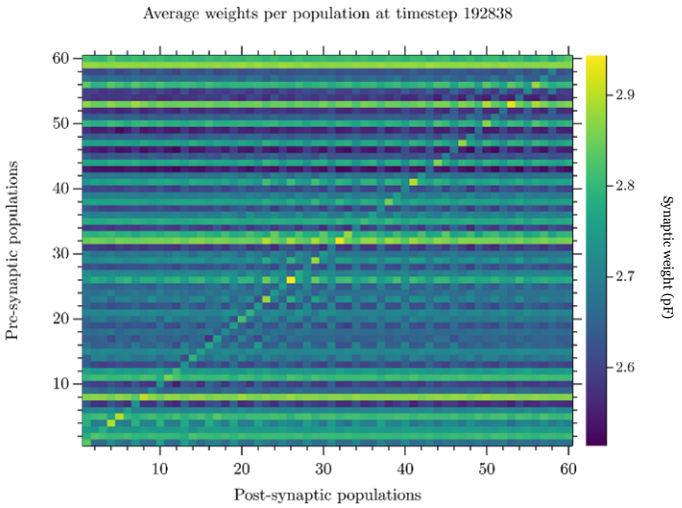
$$\text{ED} = \prod_{j=1}^{K}\left(\frac{\lambda_j}{\sum_{i=1}^{k}}\right)^{-\frac{\lambda_j}{\sum_{i=1}^{K}\lambda_i}} \tag{2.6}$$

Where K is the total number of eigenvalues and λ_j is the eigenvalue of rank j (Del Giudice, 2020).

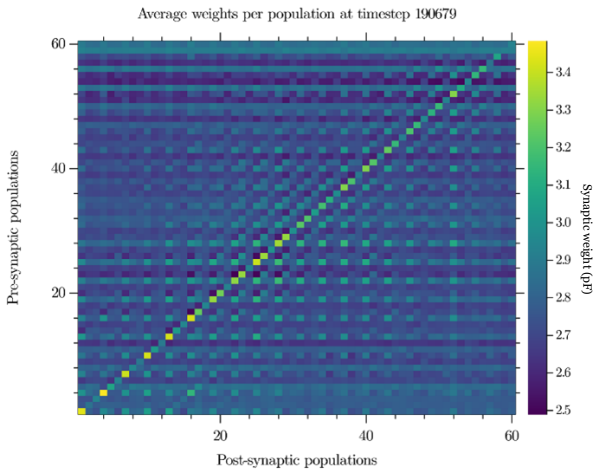
3. Results

3.1 Weight learning

As mentioned in Section 2.3.1, the original SNN model as proposed by Litwin-Kumar & Doiron (2014) uses a population coding scheme where data is encoded by a constant rate of 8kHz. The dataset used in this research however, makes use of a sparse temporal population coding scheme. This sparse coding scheme proved to not elicit enough neuronal activity for the STDP learning to alter the weights under the current parameterisation of the learning paradigm. This became apparent because neurons that are members of the same population, which are expected to have strong synaptic weights as they receive the same input (thus correlated spikes), did not have significantly stronger weights to each other than to neurons of another population. This is shown in Figure 3.1a, where the mean weight from neurons of one population to neurons of another population is shown. Here a clear diagonal is expected to appear as a result of learning. The diagonal represents the mean weights within a single population. In an attempt to make the input data evoke sufficient potential inside of the model under the current learning scheme, the pre-processing steps introduced in Section 2.2 are used. Combining the populations, as described in Section 2.2.1, did not cause the input to evoke significant activity in the model. However, this pre-processing step did significantly speed up the time required to make the weight plots shown in Figure 3.1. On the other hand, introducing the burst encoding as described in Section 2.2.2, did yield a more preferable weight distribution with the expected diagonal after training, as shown in Figure 3.1b.



(a) Weights before the burst encoding was applied



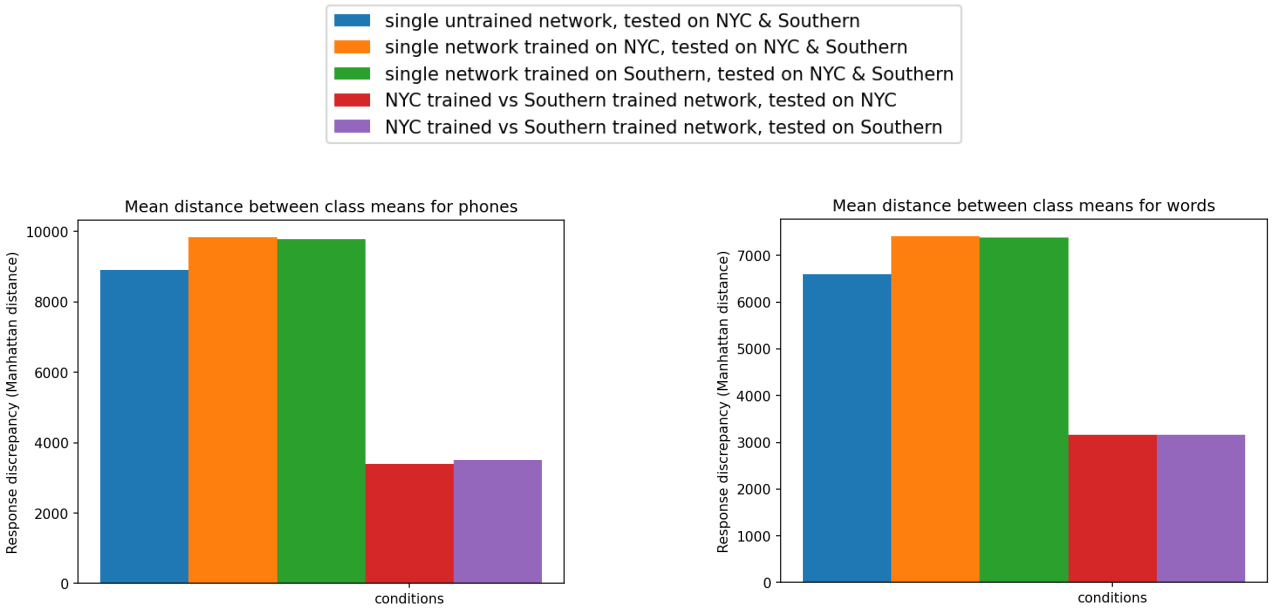
(b) Weights after the burst encoding was applied

Figure 3.1: Heatmap illustrating the mean weight of the neurons from one population to neurons of another population. The diagonal indicates the mean within population weight.

3.2 SNN comparison

To test the precision and generalisation of an SNN (how consistent its responses are to the stimuli of the same class), response discrepancies are used (Section 2.4.3). Here, the assumption is that good

generalisation of the network would yield similar responses to the same class of stimuli disregarding whether it was uttered by an NYC/male or Southern/female speaker. In this section, the results for the comparison between SNNs trained/tested on two different dialects, the NYC and Southern dialect, are given. The comparison between SNNs trained/tested on different genders (NYC males and females) yielded the same results, which are listed under Appendix Section A.8.



(a) Discrepancies between the responses to phones

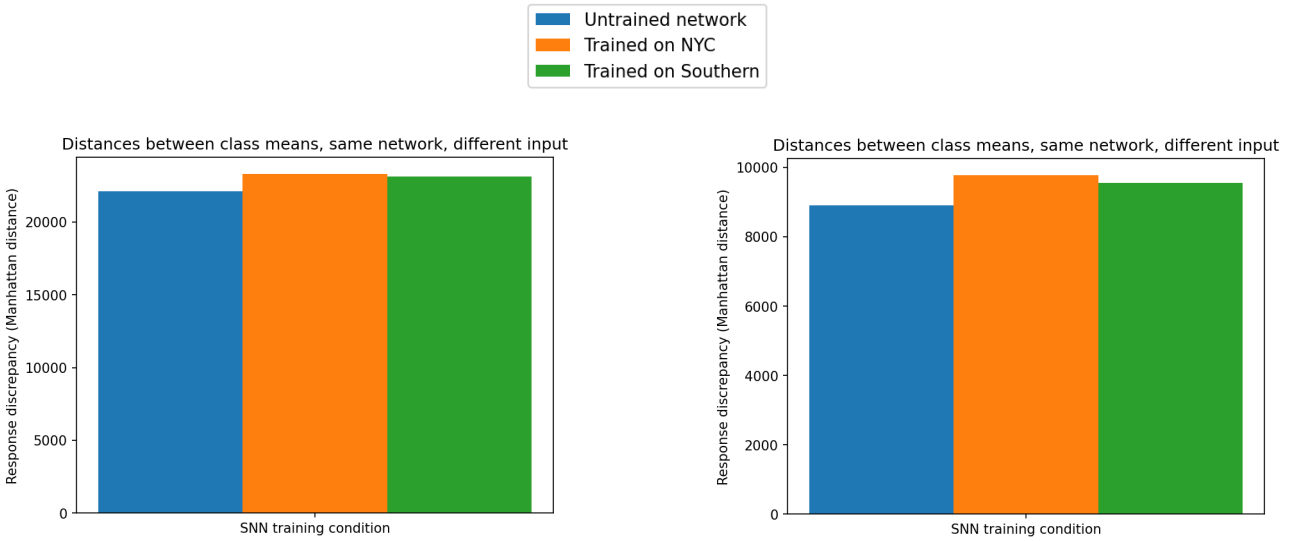
(b) Discrepancies between the responses to words

Figure 3.2: Response discrepancies for multiple testing scenarios. Each bar is the mean distance between average responses of the SNNs per stimulus class. Both the SNNs and test data are specified in the legend as well as in Figure 2.6.

First, the results in Figure 3.2 were obtained through the testing scheme described in Figure 2.6. So, in this scenario, there are three SNNs, the untrained SNN, an SNN trained on the NYC dialect and an SNN trained on the Southern dialect. For each of the bars in Figure 3.2, the exact input and which models are used can be found in Figure 2.6, by following the same colour arrow as the bar in Figure 3.2. All response discrepancies are computed by means of Formula 2.2. From Figure 3.2 it becomes apparent that learning does not improve the precision of the SNN; its responses to different inputs become more diverse after learning compared to the untrained SNN. Furthermore, from this figure, the suspicion arose that the response of the network is best determined by the input rather than the training of the network. This is because the responses of two differently trained SNNs to the same dialect (red and purple bars) was more similar than the responses of a single SNN when presented twice with a different dialect (orange and green bars). To further test this, the results in Figures 3.3 and 3.4 were gathered using the testing schemes described in Figures 2.7 and 2.8 respectively. This allows for a more precise comparison between the condition of a single SNN on two different test sets and the condition of two different SNNs on the same test set.

The first thing that stands out from Figure 3.4 is that there is a significant difference between the responses of a trained and untrained network on the same input. Furthermore, the responses of two trained networks, regardless of whether they are trained on the same but shuffled data or on another dialect, are relatively similar. However, the most important result gathered by comparing Figures 3.3 and 3.4 is that the responses of two SNNs trained on a different dialect appear to be significantly more similar than the responses of a single SNN to two different dialects. Numerical comparison indeed shows that the responses of two different SNNs to the same dialect, are 3 times more similar (the response discrepancy is 3 times smaller) than the responses of a single SNN to two different dialects. This thus means that the activity of the network is best determined by the data rather than the learn-

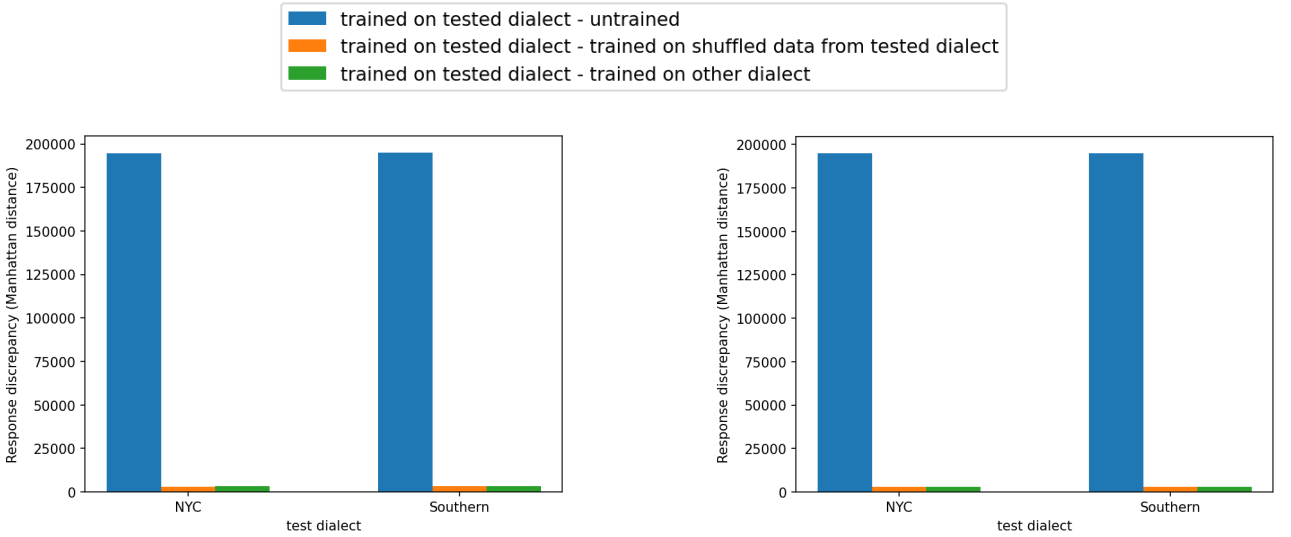
ing of the network. This suggests poor generalisation of the network's responses to phones and words.



(a) Discrepancies between the mean responses to phones

(b) Discrepancies between the mean responses to words

Figure 3.3: Discrepancies between the mean responses per stimulus class of a single SNN tested on the NYC dialect and the Southern dialect. Results are gathered following the testing scheme in Figure 2.7.

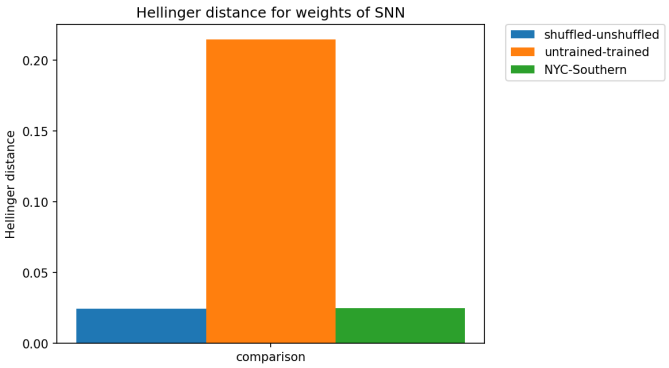


(a) Discrepancies between the mean responses to phones

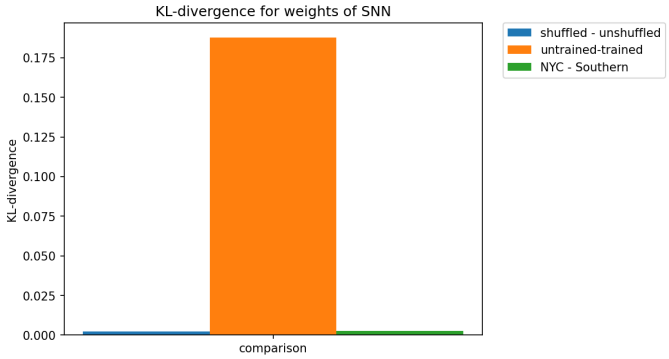
(b) Discrepancies between the mean responses to words

Figure 3.4: Discrepancies between the mean responses per stimulus class of two different SNNs tested on the same dialect. Results are gathered following the testing scheme in Figure 2.8.

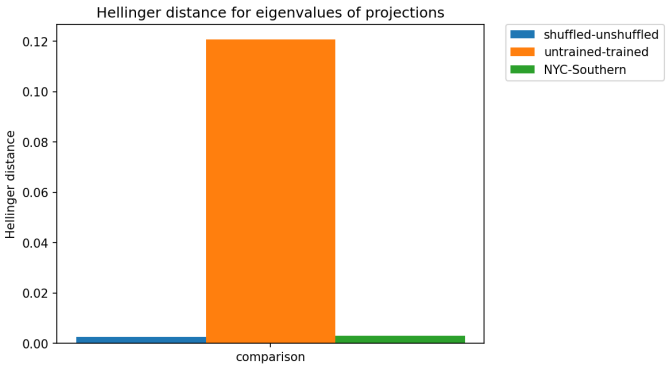
The Hellinger distances and KL-divergences used to compare SNNs, show that training two SNNs on different training sets approaches roughly the same weight distribution, this is shown in Figures 3.5a and 3.5b. The SNNs also approach roughly the same eigenvalue spectrum of their responses of the same test set, as shown in Figure 3.5c and 3.5d. This shows that when training on a shuffled train set containing the same data, as well as when training on different dialect, the SNN approaches the same distribution of weights and thus the manner in which they respond to stimuli.



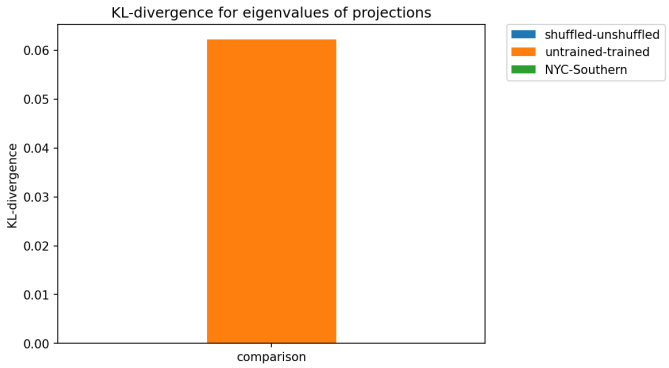
(a) Hellinger distance between weight distributions of different SNNs.



(b) KL-divergences between weight distributions of different SNNs.



(c) Hellinger distance between eigenvalue distributions of different SNN responses.



(d) KL-divergences between eigenvalue distributions of different SNN responses.

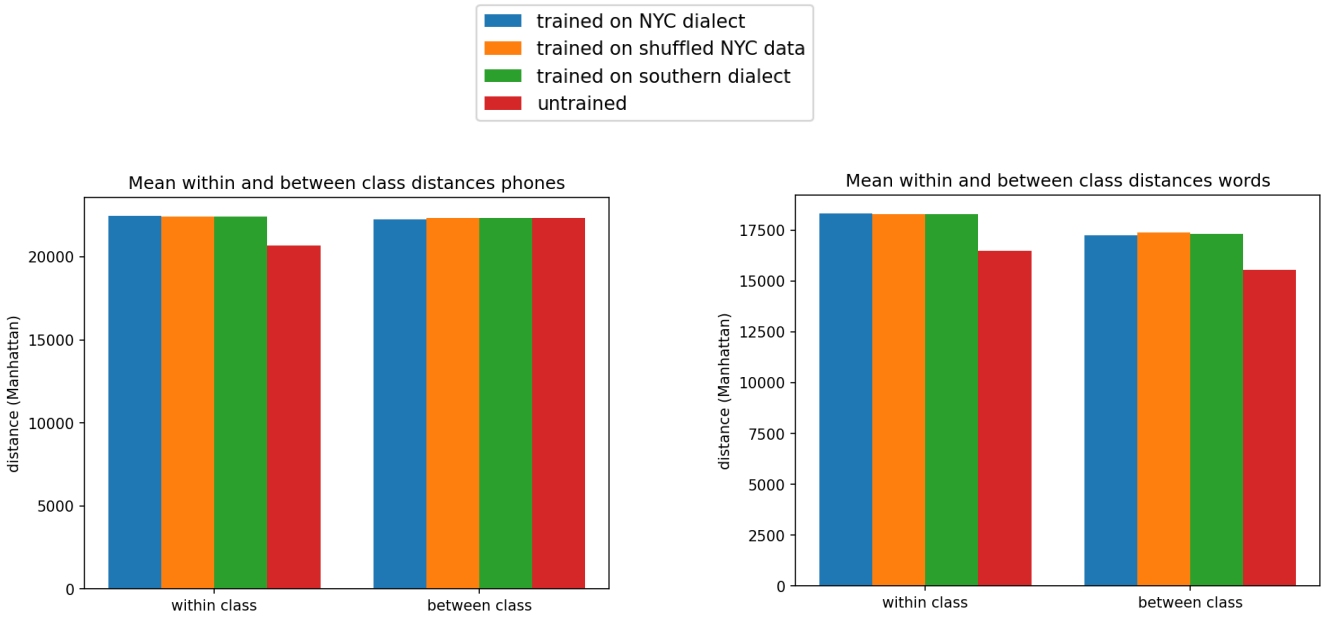
Figure 3.5: Comparison of different SNNs by their weight and eigenvalue distributions. Smaller Hellinger distances and KL-divergences indicate a higher similarity.

3.3 Separability of responses

As explained in Subsection 2.4.4, the effect of training on the separability of the responses to different stimulus classes is also analysed. From this analysis it became apparent that the separability of the responses to different stimulus classes did not improve after learning of the SNN, and in the case of phones, even worsened. This can be seen from the within and between class MADs as shown in Figure 3.6 where the within class MADs grow for the phones while the between class MADs remain unchanged, as well as from the mean classification accuracies in Figure 3.7.

3.4 Effective Dimensionality

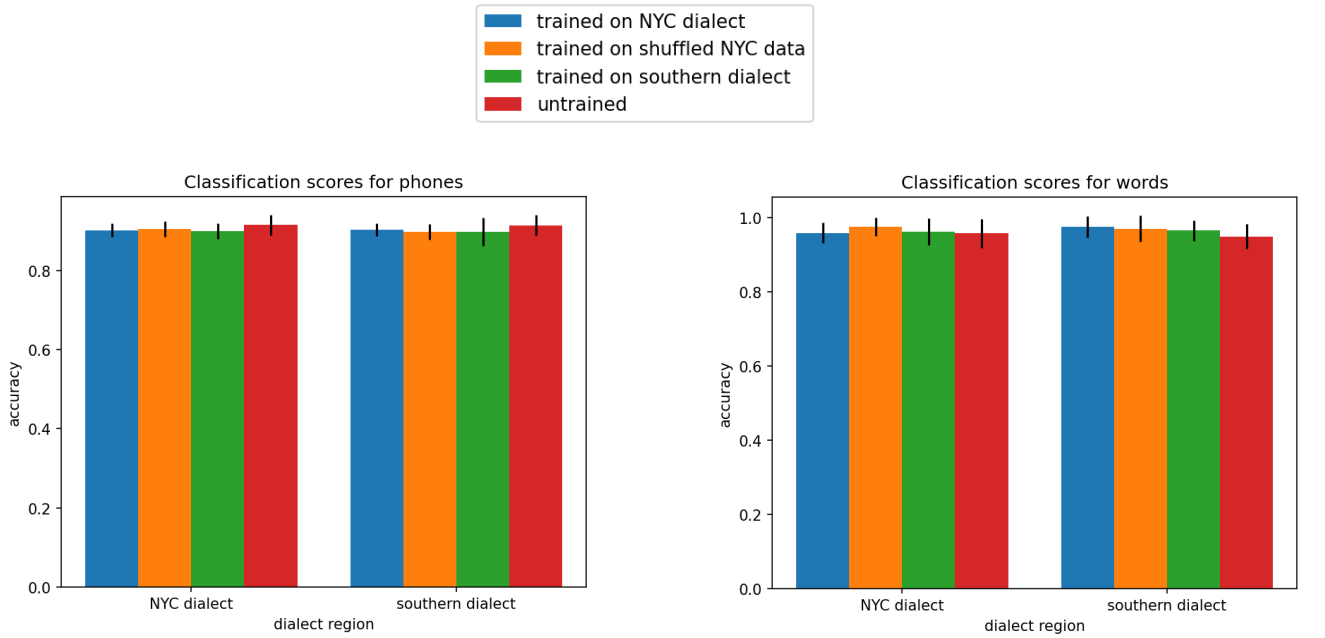
Lastly, the ED, as described in Section 2.4.5, shows that the learning of the SNN seemingly only warps the high dimensional feature space into a shape of which the variance is somewhat more uniformly distributed over the dimensions. This can be seen by comparing the EDs of the responses of trained SNNs to that of an untrained SNN, tested on the same data, shown in Figure 3.8. It is notable however that the ED is in the ballpark of $[30, 70]$ even though the dimensionality of the feature vectors are 8000. So, even though the ED becomes larger after learning of the SNN, the data is still projected onto a high dimensional flat shape rather than a shape with uniform variance over the dimensions like a ball in 3D space. The intuition behind the ED is depicted in 3.9, where the covariance matrices of PCA projections with 3 dimensions of the network responses to NYC data are depicted, before and after training the SNN.



(a) Phone within and between class distances

(b) Word within and between class distances

Figure 3.6: Within and between class distances of the responses of multiple differently trained SNNs and the untrained SNN, all tested on the same test data.



(a) Phone mean classification scores

(b) Word mean classification scores

Figure 3.7: Mean classification scores on the responses of multiple differently trained SSNs and the untrained SNN.

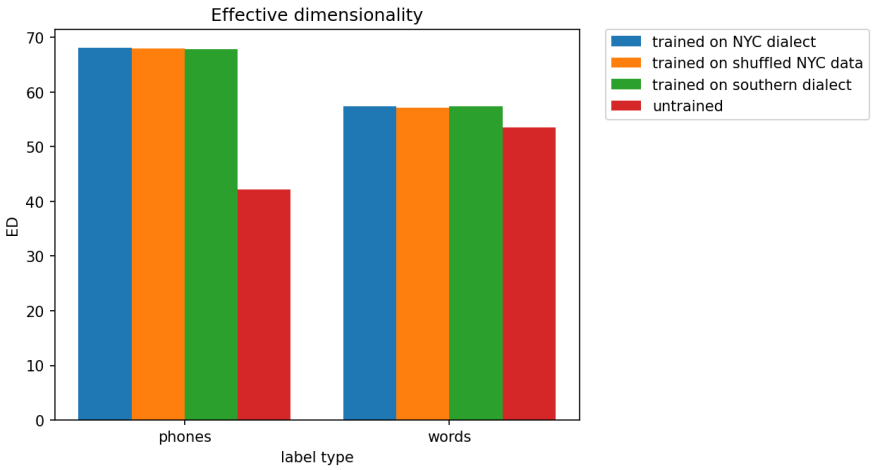
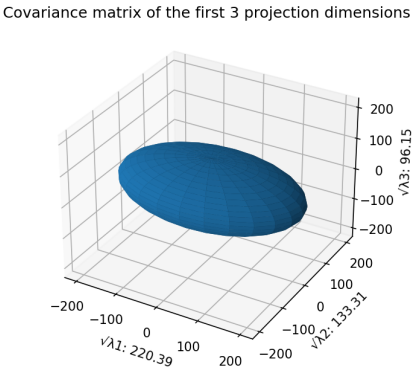
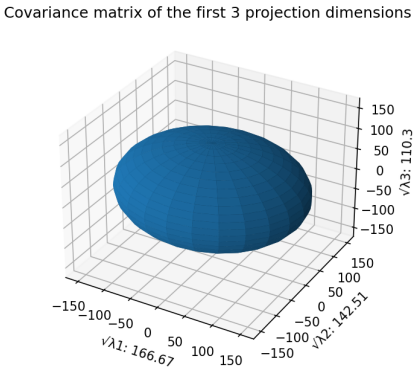


Figure 3.8: Effective dimensionality of the responses of multiple trained SNNs and the untrained SNN on the same data.



(a) Untrained SNN responses covariance matrix of first 3 PCs



(b) Trained SNN responses covariance matrix of first 3 PCs

Figure 3.9: Covariance matrices of the first 3 PCs after PCA projection of the responses of an untrained and trained SNN to phones, uttered by NYC speakers.

4. Discussion and Related Work

In this project, we expanded on the model proposed by Litwin-Kumar & Doiron (2014) in an attempt to improve it with regard to the biological plausibility of the input. Furthermore, the original SNN as proposed by Litwin-Kumar & Doiron (2014), showed that such models can learn stable assemblies (memories) under the proposed mechanisms (Litwin-Kumar & Doiron, 2014). However, we are interested in using the SNN in a speech recognition pipeline and learn about the simulated area in the brain. In particular, I am interested in what representations are learned.

The assumption was that good generalisation of the network would yield similar responses to the same class of stimuli disregarding whether it was uttered by an NYC/male or Southern/female speaker. Given this assumption and the fact that the response of a network is best determined by the data rather than its training, it appears that the SNN learns no generalisation of its responses to phones and words. These results also suggest that the network is more sensitive to the physical properties of the stimulus rather its phonological content, such as the phone or word. This would be sensible as the simulated area is a very low-level area that receives input directly from the cochlea, only consisting of information about the intensity of frequency bands. It could be the case that this simulated region first processes this very raw data before it is passed on to a region that uses this activity to convert it into the representations of phones.

What the exact physical features are to which the network is sensitive can not be shown by the results in this project. Learning about whether the SNN learned other representations requires extending the Spike-TIMIT data set to include labels and intervals of acoustic-phonetic features, allophones and phonemes in order to test whether these speech units are learned by the SNN. Currently, this analysis is limited to only the representations of phones and words, where the words are very unlikely to be learned by the SNN as it receives very low-level input and thus simulates an area in the prelexical processing phase of speech recognition. Lastly, even though we succeeded in making the Spike-TIMIT (Pan et al., 2020) data set evoke significant activity in the model, this was done by adding extra spikes, be it in a biologically plausible way, and by giving a much higher synaptic weight to the input data than to the background activity. This then means that a distinction is being made between what input spikes are part of the input signal and what input spikes are background activity. For the area which we simulate, this may not be completely biologically plausible. However, one could argue that the effect that attention might have on speech processing justifies a distinction between the weights assigned to signal and noise. Future research could shine a light on a more plausible way of balancing these parameters such that input spikes of the data do not necessarily instantly cause spiking within the network.

A big inspiration for this thesis was the work by Dong et al. (2018), who researched receptive fields of SNNs on speech data. However, in that work, they made use of a convolutional layer (Dong et al., 2018) whereas we did not. Furthermore, it has been shown that in human speech recognition, both feedforward and feedback paths are present (Tang & Suga, 2008). This means that the purely feed forward model proposed by Dong et al. (2018) is in that regard less biologically plausible than the fully recurrent SNN used in this thesis. Nevertheless, we did take inspiration from the 'three stages of model evaluation' that was used in this work. Furthermore, the claims about receptive fields made in the work by Dong et al. (2018) should be verified as they interpret the weights as being analogous to receptive fields. This is typically not the case in temporal data, an example of this is the method of

Common Spatial Patterns (CSP) in the domain of signal processing where the weights only tell half of the story. The weights indicate which features are amplified or inhibited, but this could be a result of filtering out noise for example and should thus be weighted by the (covariance) of the data. However, the weights might give a reasonably approximation of the receptive fields.

5. Conclusion

In this thesis, I have investigated if learning of an SNN on speech data can lead to meaningful speech unit representations and if so, what representations are learned by the SNN. The results however, indicate that it is not the case that the hypothesised representation of phones was learned by the network. In addition, learning of the SNN under the current learning paradigm seemingly only warps the responses to presented stimuli into a shape of which the variance is more uniformly distributed over the dimensions and does not improve the separability of the responses to different phones or words. All results together suggest that the responses of the SNN are sensitive to the physical characteristics of the signal rather than to its phonological context. However, further research is required to make a definitive claim about the types of representations or stimuli to which this network is most sensitive. In conclusion, the training of an SNN on spoken speeh data under the current learning paradigm did not yield any generalisation properties of the network's responses to the representations of phones or words.

References

Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory* (pp. 420–434).

Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., ... others (2015). Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, *34*(10), 1537–1557.

Aston Zhang, M. L., Zachary C. Lipton, & Smola, A. J. (2021). Dive into deep learning. In (p. 32).

Badel, L., Lefort, S., Brette, R., Petersen, C. C., Gerstner, W., & Richardson, M. J. (2008). Dynamic iv curves are reliable predictors of naturalistic pyramidal-neuron voltage traces. *Journal of Neurophysiology*, *99*(2), 656–666.

Bi, G.-q., & Poo, M.-m. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience*, *18*(24), 10464–10472.

Bowers, J. S., Kazanina, N., & Andermane, N. (2016). Spoken word identification involves accessing position invariant phoneme representations. *Journal of Memory and Language*, *87*, 71–83.

Brette, R., & Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of neurophysiology*, *94*(5), 3637–3642.

Clopath, C., Büsing, L., Vasilaki, E., & Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based stdp with homeostasis. *Nature neuroscience*, *13*(3), 344.

Del Giudice, M. (2020). Effective dimensionality: A tutorial. *Multivariate behavioral research*, 1–16.

Dong, M., Huang, X., & Xu, B. (2018). Unsupervised speech recognition through spike-timing-dependent plasticity in a convolutional spiking neural network. *PloS one*, *13*(11), e0204596.

Fiete, I. R., Senn, W., Wang, C. Z., & Hahnloser, R. H. (2010). Spike-time-dependent plasticity and heterosynaptic competition organize networks to produce long scale-free sequences of neural activity. *Neuron*, *65*(4), 563–576.

Garofolo, J. S. (1993). Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993*.

Gütig, R., & Sompolinsky, H. (2009). Time-warp–invariant neuronal processing. *PLoS Biol*, *7*(7), e1000141.

Liberman, M. C. (1982). The cochlear frequency map for the cat: Labeling auditory-nerve fibers of known characteristic frequency. *The Journal of the Acoustical Society of America*, *72*(5), 1441–1449.

Litwin-Kumar, A., & Doiron, B. (2014). Formation and maintenance of neuronal assemblies through synaptic plasticity. *Nature communications*, *5*(1), 1–12.

Maddieson, I. (2013). Voicing and gaps in plosive systems. *The world atlas of language structures online*.

Mitterer, H., Reinisch, E., & McQueen, J. M. (2018). Allophones, not phonemes in spoken-word recognition. *Journal of Memory and Language*, *98*, 77–92.

Myers, E., Johns, A., Earle, S., & Xie, X. (2017, 01). The invariance problem in the acquisition of non-native phonetic contrasts: From instances to categories: Neurocognitive and behavioural approaches.. doi: 10.1515/9783110422658-004

Nikulin, M. (1994). *Hellinger distance.* EMS Press. Retrieved from `https://encyclopediaofmath.org/index.php?title=Hellinger_distance`

Oswald, A.-M. M., Doiron, B., & Maler, L. (2007). Interval coding. i. burst interspike intervals as indicators of stimulus intensity. *Journal of neurophysiology*, *97*(4), 2731–2743.

Pan, Z., Chua, Y., Wu, J., Zhang, M., Li, H., & Ambikairajah, E. (2020). An efficient and perceptually motivated auditory neural encoding and decoding algorithm for spiking neural networks. *Frontiers in neuroscience*, *13*, 1420.

Pisoni, D. B. (1985). Speech perception: Some new directions in research and theory. *The Journal of the Acoustical Society of America*, *78*(1), 381–388.

Renart, A., Song, P., & Wang, X.-J. (2003). Robust spatial working memory through homeostatic synaptic scaling in heterogeneous cortical networks. *Neuron*, *38*(3), 473–485.

Taleb, N. N. (1970, Jan). *What scientific idea is ready for retirement?* Retrieved from `https://web.archive.org/web/20140116031136/http://www.edge.org/response-detail/25401`

Tang, J., & Suga, N. (2008). Modulation of auditory processing by cortico-cortical feed-forward and feedback projections. *Proceedings of the National Academy of Sciences*, *105*(21), 7600–7605.

Thorpe, S. J. (1990). Spike arrival times: A highly efficient coding scheme for neural networks. *Parallel processing in neural systems*, 91–94.

Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C., & Gerstner, W. (2011). Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science*, *334*(6062), 1569–1573.

Wu, S., Amari, S.-i., & Nakahara, H. (2002). Population coding and decoding in a neural field: a computational study. *Neural Computation*, *14*(5), 999–1026.

Yi, H. G., Leonard, M. K., & Chang, E. F. (2019). The encoding of speech sounds in the superior temporal gyrus. *Neuron*, *102*(6), 1096–1110.

Zeldenrust, F., Wadman, W. J., & Englitz, B. (2018). Neural coding with bursts—current state and future perspectives. *Frontiers in computational neuroscience*, *12*, 48.

Appendix

This appendix was largely created in cooperation with my fellow students Fleur Brandsen, Léonie Wagner and Joost Kasper. The basis of the model was a joint effort to make it feasible for all of us to complete a project of this magnitude.

A.1 Acoustic-phonetic features

Acoustic-phonetic, or sub phonetic, features describe the underlying acoustic properties and articulatory gestures that generate speech sounds (Yi et al., 2019). To give an example of these acoustic-phonetic features, you can have plosives (also known as 'a stop' (Maddieson, 2013)) and fricatives. Plosives completely close the airflow followed by quickly releasing the built-up pressure. To give an example of a phoneme described by sub phonetic features: a bilabial (requires both lips), voiceless (no vibration in vocal cords), plosive is the phoneme /p/ (Yi et al., 2019). Whereas fricatives are produced by restricting the airflow, while still allowing some air to pass through with friction. An example of a phoneme that is a labio-dental (requires a lip and teeth), voiceless fricative is the phoneme /f/ (Yi et al., 2019).

A.2 Weights and Learning

A.2.1 Weight values

The parameters used come from the model as described by Litwin-Kumar & Doiron (2014), because we wanted to recreate their dynamics. Below, their parameters are given, together with a corresponding explanation about the internal structure and working of their model. The initial weight values are given in Table A.1. Of these weights, only the weights going to excitatory neurons can be modified. The changing of weights is done through STDP if the presynaptic neuron is excitatory, as described in Section A.2.2. If the presynaptic neuron is inhibitory the weight change is by the means of iSTDP, as described in Section A.2.3. These synaptic weights are bound by the values given in Table A.2. The probability a neuron is connected to any other neuron is 0.2. If there is no synaptic connection, the weight is set to 0 in the weight matrix and is never altered, regardless if the postsynaptic neuron is excitatory.

From \ To	Excitatory	Inhibitory
Excitatory	2.86 pF	48.7 pF
Inhibitory	1.27 pF	16.2 pF

Table A.1: Initial synaptic weight values.

From \ To	Excitatory
Excitatory	[1.78 pF, 21.4 pF]
Inhibitory	[48.7 pF, 243 pF]

Table A.2: Lower and upper bounds for the synaptic weights going to excitatory neurons.

Description	**Value**
Probability for synapse connections	0.2
Probability of belonging to a population	0.1

Table A.3: Connectivity probabilities of the network

A.2.2 STDP

The learning rule used for connections between excitatory neurons is STDP (Clopath et al., 2010; Litwin-Kumar & Doiron, 2014). This rule alters only the synaptic weights that go from an excitatory neuron to another excitatory neuron. These weights cannot just have any value, but are restricted to the values given in Table A.2. Formula A.1 models the dynamics of STDP, where i and j refer to the presynaptic neuron and postsynaptic neuron respectively.

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}J_{ij}^{EE}(t) = &- A_{\mathrm{LTD}}{}_j^E(t)R\left(u_i^E(t) - \theta_{\mathrm{LTD}}\right) \\ &+ A_{\mathrm{LTP}}x_j^E(t)R\left(V_i^E(t) - \theta_{\mathrm{LTP}}\right)R\left(v_i^E(t) - \theta_{\mathrm{LTD}}\right)\end{aligned} \tag{A.1}$$

As described in Litwin-Kumar & Doiron (2014), in this formula J_{ij}^{EE} is the synaptic weight from excitatory neuron j to excitatory neuron i in pF and R is a linear-rectifying function ($R(x) = 0$ if $x < 0, R(x) = x$ otherwise). x_j^E is the result of low-pass filtering spike train s_j^E (Section A.3). The time constant used is τ_x. The membrane voltage V_i^E, see Formula A.5, is also low-pass filtered twice. Once with time constant τ_u to obtain u_i^E and once with time constant τ_v to obtain v_i^E. The values of these parameters can be found in Table A.4.

Symbol	Description	Value
A_{LTD}	Long-term depression (LTD) strength	0.0008 pA mV^{-1}
A_{LTP}	Long-term potentiation (LTP) strength	0.0014 pA mV^{-2}
θ_{LTD}	Threshold to recruit LTP	-70 mV
θ_{LTP}	Threshold to recruit LTP	-49 mV
τ_u	Time constant of low-pass filtered membrane voltage (for LTD)	10 ms
τ_v	Time constant of low-pass filtered membrane voltage (for LTP)	7 ms
τ_x	Time constant of low-pass filtered spike train (for LTP)	15 ms
τ_y	Time constant of low-pass filtered spike train	20 ms
η	Synaptic plasticity learning rate	1 pA
r_0	Target firing rate	3 Hz

Table A.4: Parameters used for updating the excitatory weights (Litwin-Kumar & Doiron, 2014).

For each neuron its incoming synaptic weights are normalised by a homeostatic normalisation (Litwin-Kumar & Doiron, 2014). It means that for a neuron all weights going towards that neuron are scaled such that the sum of these weights stays at a predefined value. This normalisation is happening every 20ms for all neurons (Litwin-Kumar & Doiron, 2014; Renart et al., 2003; Fiete et al., 2010). Formula A.2 describes this mechanism. N_i^E is the number of connections that go from any neuron to neuron i.

$$J_{ij}^{EE}(t) \leftarrow J_{ij}^{EE}(t) - \left(\left(\sum_j J_{ij}^{EE}(t) - J_{ij}^{EE}(0) \right) / N_i^E \right) \tag{A.2}$$

A.2.3 iSTDP

To change the synaptic weights going from inhibitory neurons to excitatory neurons iSTDP is used (Vogels et al., 2011). The weights are bound by the values given in Table A.2, just as with STDP. The dynamics of the synapse from excitatory neuron j to inhibitory neuron i is given by Formula A.3 (Litwin-Kumar & Doiron, 2014).

$$\begin{aligned} J_{ij}^{EI} &\leftarrow J_{ij}^{EI} + \eta \left(y_i^E(t) - 2r_0\tau_y \right) && \text{if the presynaptic inhibitory neuron fired} \\ J_{ij}^{EI} &\leftarrow J_{ij}^{EI} + \eta y_j^I(t) && \text{if the postsynaptic excitatory neuron fired} \end{aligned} \tag{A.3}$$

In this formula y_i^X is obtained by a low-pass filter applied to the spike train s_i^X (Section A.3). For this τ_y was used as time constant. The rate r_0 is the target firing rate of the postsynaptic neuron which the inhibitory plasticity attempts to maintain. These parameters can be found in Table A.4.

A.3 Spike trains

Spike trains are a concatenation of times at which a neuron spikes. The spike train of neuron i of type X (excitatory E or inhibitory I) is denoted as $s_i^X(t)$ and is given by Formula A.4 (Litwin-Kumar &

Doiron, 2014).

$$s_i^X(t) = \sum_{k=1}^{n} \delta\left(t - t_{i,k}^X\right) \tag{A.4}$$

Where δ denotes the Dirac delta function, and $t_{i,1}^X ... t_{i,n}^X$ are the times at which this neuron i spiked.

A.4 Membrane potential dynamics

The network has $4,000$ excitatory and $1,000$ inhibitory neurons, denoted by N^E and N^I respectively. The type of neuron (E or I) is indicated by superscript X, whereas the index of the neuron is denoted by subscript i. The membrane voltage of a neuron is given by Formula A.5 (Litwin-Kumar & Doiron, 2014) (Clopath et al., 2010). However, when a neuron spikes, its membrane potential is reset to V_{re} and fixed for a refractory period (τ_{abs}).

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}V_i^{\mathrm{X}}(t) =& \frac{1}{\tau^X}\left(E_L^X - V_i^X(t) + \Delta_T^X \exp\left(\frac{V_i^X(t) - V_{T,i}^X(t)}{\Delta_T^X}\right)\right) \\ &+ \frac{g_i^{XE}(t)}{C}\left(E^E - V_i^X(t)\right) + \frac{g_i^{XI}(t)}{C}\left(E^I - V_i^X(t)\right) - \frac{w_i^X(t)}{C}\end{aligned} \tag{A.5}$$

All the parameters used can be found in Table A.5.

All excitatory neurons are modelled as exponential integrate-and-fire neurons (Litwin-Kumar & Doiron, 2014). This type of neuron has an adaptation current and adaptive threshold (Brette & Gerstner, 2005; Badel et al., 2008). All inhibitory neurons are designed as non-adapting integrate-and-fire neurons (Litwin-Kumar & Doiron, 2014). The adaptive threshold is given by Formula A.6 and the adaptation current is given by Formula A.7 (Litwin-Kumar & Doiron, 2014). However, when an excitatory neuron spikes, its threshold is reset to $V_T + A_T$ (given in Table A.5). The threshold for inhibitory neurons always remains V_T (Table A.5). For the adaptation current, if excitatory neuron i spiked, b_w was added to w_i^E (Table A.5).

$$\frac{\mathrm{d}}{\mathrm{d}t}V_{T,i}^E(t) = \frac{1}{\tau_T}\left(V_T - V_{T,i}^E(t)\right) \tag{A.6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}w_i^E(t) = \frac{1}{\tau_w}\left(a_w\left(V_i^E(t) - E_L^E\right) - w_i^E(t)\right) \tag{A.7}$$

Where all parameters of Formula A.7 can be found in Table A.5.

A.5 Synaptic conductances

Formula A.8 shows the total conductance of neuron i that belongs to population X (Litwin-Kumar & Doiron, 2014). Where Y $\in$ {E,I}, $F^Y(t)$ comes from Formula A.9, and the $*$ denotes convolution.

$$g_i^{XY}(t) = F^Y(t) * \left(J_{\text{ext}}^{XY} s_{i,\,\text{ext}}^{XY}(t) + \sum_j J_{ij}^{XY} s_i^Y(t)\right) \tag{A.8}$$

Formula A.9 describes the synaptic kernel for input from population Y.

$$F^Y(t) = \frac{1}{\tau_d^Y - \tau_r^Y}\left(e^{-t/\tau_d^Y} - e^{-t/\tau_r^Y}\right) \tag{A.9}$$

Where all parameters can be found in Table A.5.

Symbol	Description	Value
τ^E	E neuron resting membrane time constant	20 ms
τ^I	I neuron resting membrane time constant	20 ms
E_L^E	E neuron resting potential	-70 mV
E_L^I	I neuron resting potential	-62 mV
Δ_T^E	E neuron exponential integrate-and-fire slope factor	2 mV
C	Capacitance	300 pF
E^E	E reversal potential	0 mV
E^I	I reversal potential	0 mV
E^I	I reversal potential	-75 mV
V_T	Threshold potential	-52 mV
A_T	Post spike threshold potential increase	10 mV
V_{re}	Reset potential	-60 mV
τ_T	adaptive threshold time scale	30 ms
τ_{abs}	Absolute refractory period	1 ms
a_w	Subthreshold adaptation	4 nS
b_w	Spike-triggered adaptation	0.805 pA
τ_w	Spike-triggered adaptation time scale	150 ms
J_{min}^{EE}	Minimum E to E synaptic weight	1.78 or 100 pF *
J_{min}^{IE}	Minimum I to E synaptic weight	1.27 pF
τ_r^E	Rise time for E synapses	1 ms
τ_d^E	Decay time for E synapses	6 ms
τ_r^I	Rise time for I synapses	0.5 ms
τ_d^I	Decay time for I synapses	2 ms

Table A.5: Parameters used for membrane potential dynamics (Litwin-Kumar & Doiron, 2014).
* In the original model by Litwin-Kumar & Doiron (2014) the minimal E to E synaptic weight was 1.78 pF. This value is still used to model the background activity of the excitatory neurons within the network. However, in order for the spike input to activate the network with a sufficient signal-to-noise ratio this value was changed to 100 pF for the processing of the input signal.

Furthermore, $s_{i,\,\text{ext}}^{XY}(t)$ denotes the spike train of the external input (the background activity and data), whereas s_i^Y is the spike train given by Formula A.4. J_{ext}^{XY} denotes the synaptic weight strength of the external input. This means that excitatory neurons received background activity spikes weighted by $J_{\text{ext}}^{XY} = J_{\text{min}}^{EE}$ (Table A.5) but spikes from the data input are weighted by $J_{\text{ext}}^{XY} = 100pF$. Inhibitory neurons received background activity spikes weighted by $J_{\text{ext}}^{XY} = J_{\text{min}}^{IE}$ (Table A.5).

A.6 Combining input populations

The implementation is somewhat cumbersome because you do not want to mix neurons that encode different frequencies. Since the 620 original neurons are split into groups of 31 neurons encoding for a single frequency band, the combining is done within that group of 31 neurons. This means that if the 31 neurons are not perfectly dividable over the number of neurons you want to combine, some new populations will consist of more neurons than the others. The implementation is given by the following

code block. Where *Spiketimes* is a vector of 620 lists, one for external each neuron, where each list contains the times at which that neuron spikes during the presentation of a stimulus.

```
"""
        Unify frequencies bin
        Input: spiketimes array with elements of size 620 elements
        Return array with sorted spikes in less classes
"""
function resample_spikes(;spiketimes::Vector{Spiketimes},n_feat)
        for s in 1:length(spiketimes)
                spiketimes[s] = _resample_spikes(spiketimes=spiketimes[s],
                n_feat=n_feat)
        end
        return spiketimes
end

function _resample_spikes(;spiketimes::Spiketimes, n_feat)
    # If we don't reduce the bins
    if n_feat == 1
        return spiketimes
    elseif n_feat > 11  n_feat < 1
        println("WARNING; you are crazy, returning original spiketimes")
        return spiketimes
    end

    REQUENCIES = 20

    old_bins = convert(Int64, length(spiketimes)/FREQUENCIES)
    @assert (old_bins==31) "WARNING: old_bins != 31, function may not work"
    new_bins = round(Int, old_bins/n_feat - 0.1)
    add_last = 0
    if n_feat*new_bins < 31
        add_last = 31-n_feat*new_bins
    end

        new_spikes = map(x->Vector{Float64}(),1:new_bins*FREQUENCIES)

    for freq in 1:FREQUENCIES
        old_freq = (freq-1)*old_bins
        new_freq = (freq-1)*new_bins
        for new_bin in 1:new_bins
            last_bin = new_bin*n_feat <32 ? new_bin*n_feat : 31
            if new_bin == new_bins
                last_bin += add_last
            end
            bins = 1+(new_bin-1)*n_feat : last_bin
            for old_bin in bins
                push!(new_spikes[new_bin+new_freq], spiketimes[old_bin+old_freq]...)
            end
        end
    end
        return sort!.(new_spikes)
end
```

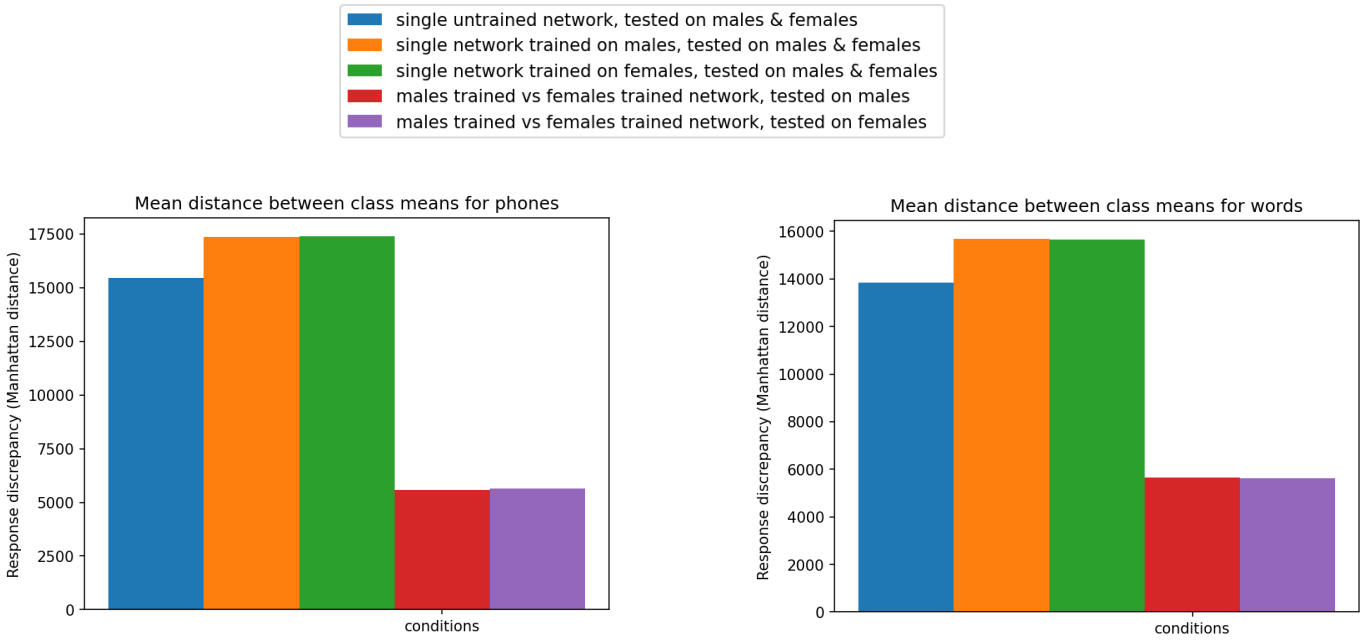
A.7 confidence interval

The confidence intervals of 95% of the mean classification scores were computed by means of:
lower bound $= \mu - t^* \frac{\sigma}{\sqrt{n}}$
upper bound $= \mu + t^* \frac{\sigma}{\sqrt{n}}$
Where $t^* = 2.262$ for a confidence interval of 95% with a sample size of $n = 10$.

A.8 Results gender comparison

As mentioned in Section 3, the results were nearly identical for both the dialect and gender comparison. Here the results are given for the gender comparison.

Training is expected to also not offer any generalisation of the responses of the network to phones or words in the case of the gender comparison. To show this, the testing schemes in Figures 2.6, 2.7 and 2.8 were used to obtain the results in Figures A.1, A.2 and A.3 respectively. From these figures it becomes apparent that learning does not improve the precision of the SNN; its responses to different inputs become more diverse after learning compared to the untrained SNN. In addition, the responses of two differently trained SNNs to the same gender was, by numerical comparison, 3 times more similar than the responses of a single SNN when tested on each gender. These plots indeed suggest that the network learned no generalisation properties for its responses to phones or words, as the response is best determined by the data rather than the network.



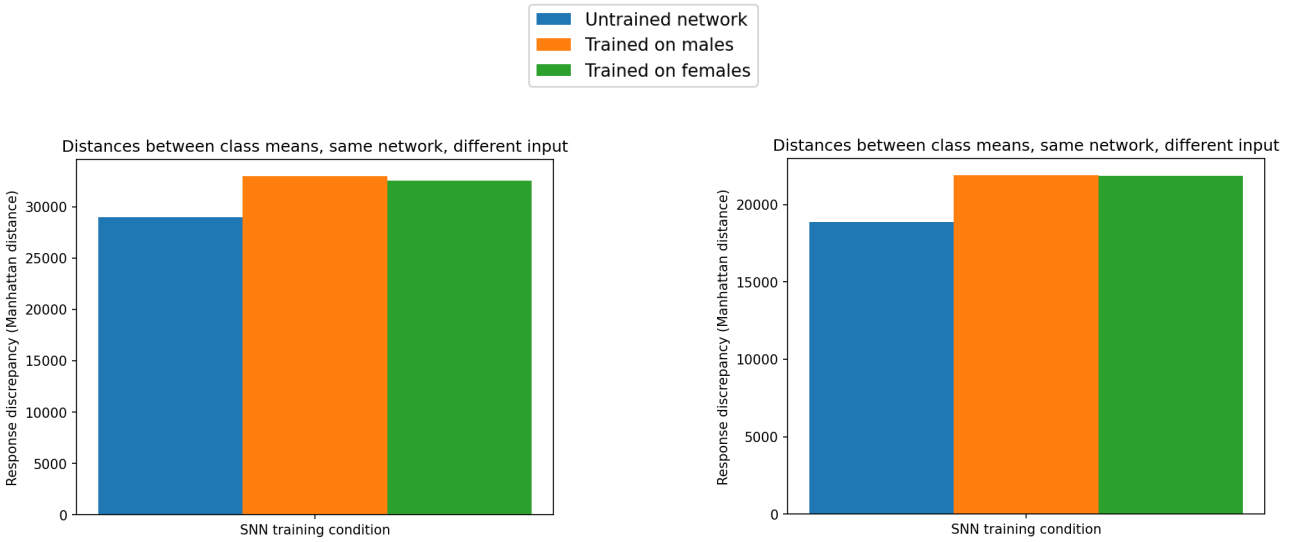
(a) Discrepancies between the responses to phones

(b) Discrepancies between the responses to words

Figure A.1: Response discrepancies for multiple testing scenarios. Each bar is the mean distance between average responses of the SNNs per stimulus class. Both the SNNs and test data are specified in the legend as well as in Figure 2.6.

Next, as shown in Figures A.4 and A.5, training on a single gender did not improve the separability of the network's responses to phones and words either. In the case of phones, the separability of the responses of the network even worsens.

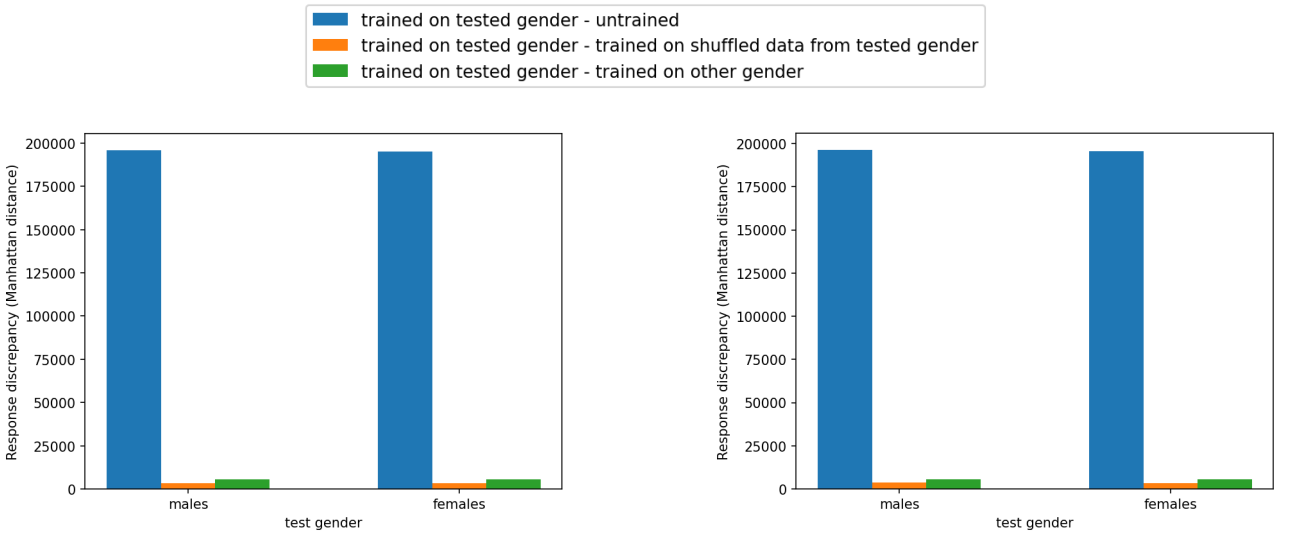
From Figure A.6 it becomes apparent that also in this case, training on the same, but shuffled, data, or training on another gender, the network approaches the same weight distribution and eigenvalue distribution of the responses to the same test data. This suggest that training on different data sets approaches the same network.



(a) Discrepancies between the mean responses to phones

(b) Discrepancies between the mean responses to words

Figure A.2: Discrepancies between the mean responses per stimulus class of a single SNN tested on both the genders. Results are gathered following the testing scheme in Figre 2.7.

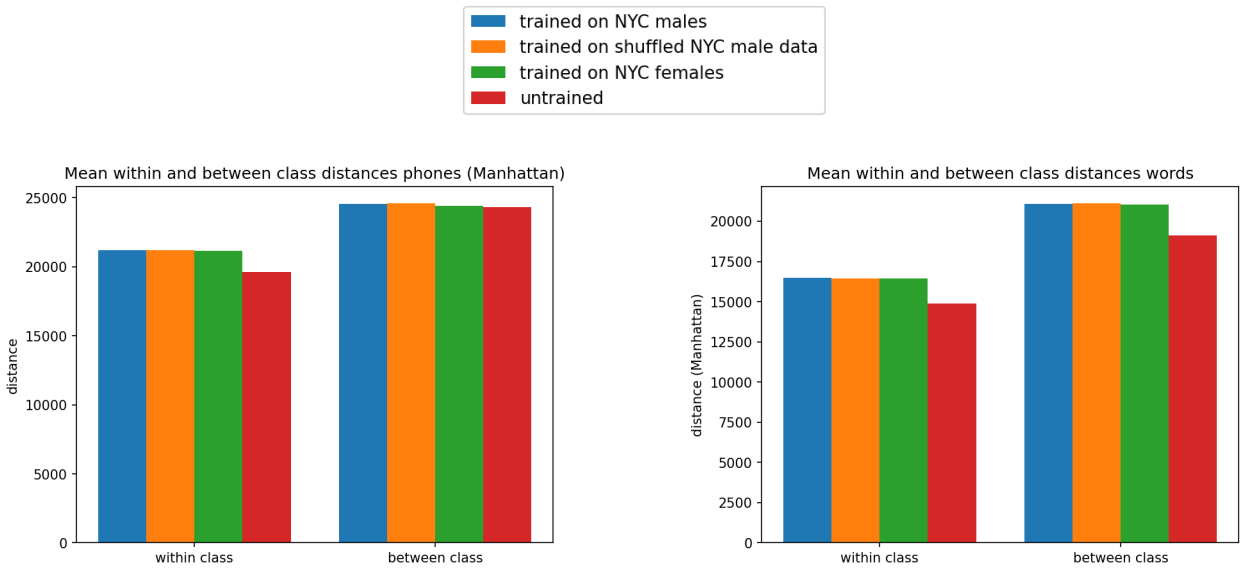


(a) Discrepancies between the mean responses to phones

(b) Discrepancies between the mean responses to words

Figure A.3: Discrepancies between the mean responses per stimulus class of two different SNNs tested on the same gender. Results are gathered following the testing scheme in Figure 2.8.

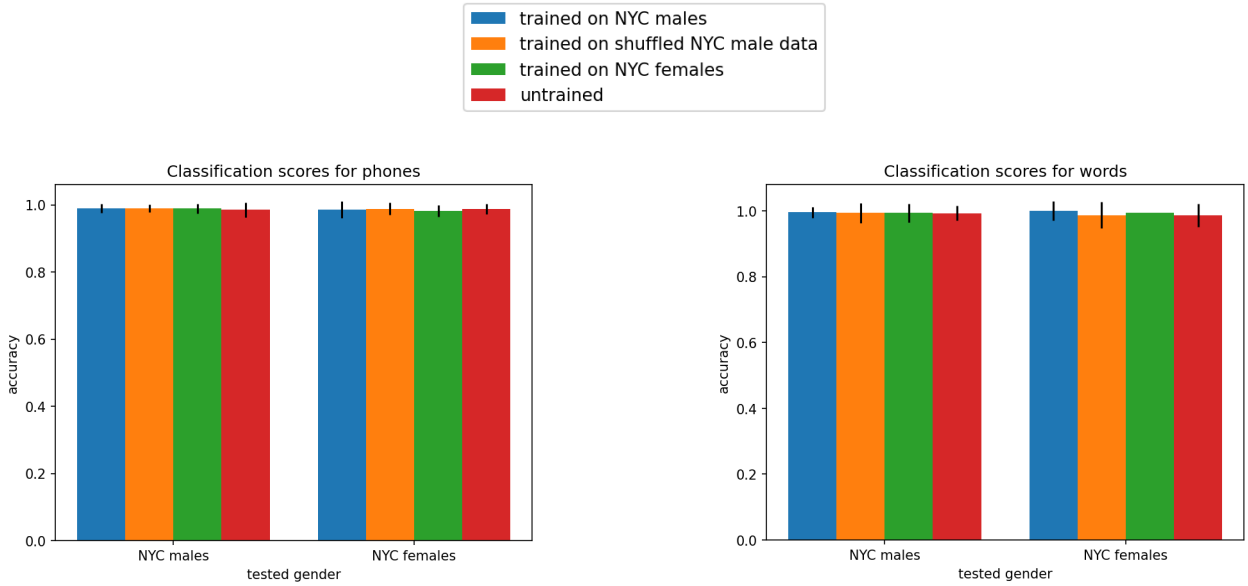
Lastly, the training seemingly only warps the high dimensional feature space into a more uniformly distributed shape as can be seen from comparing the EDs of the responses of trained SNNs to that of an untrained SNN, tested on the same data, shown in Figure A.7. The intuition behind the ED is depicted in A.8, where the covariance matrices of PCA projections with 3 dimensions of the responses of an untrained a trained SNN tested on NYC males data are depicted.



(a) Phone within and between class distances

(b) Word within and between class distances

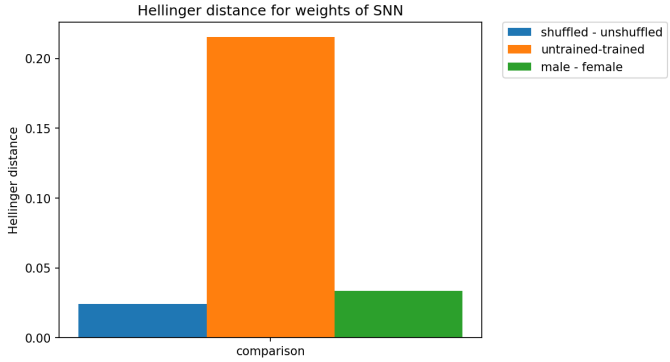
Figure A.4: Within and between class distances of the responses of multiple trained SNNs and the untrained SNN, all tested on the same test data.



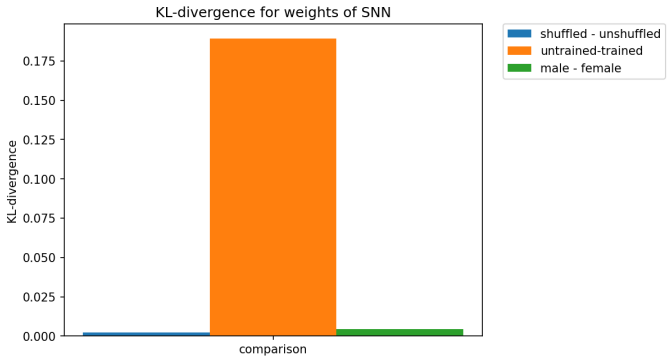
(a) Phone mean classification scores

(b) Word mean classification scores

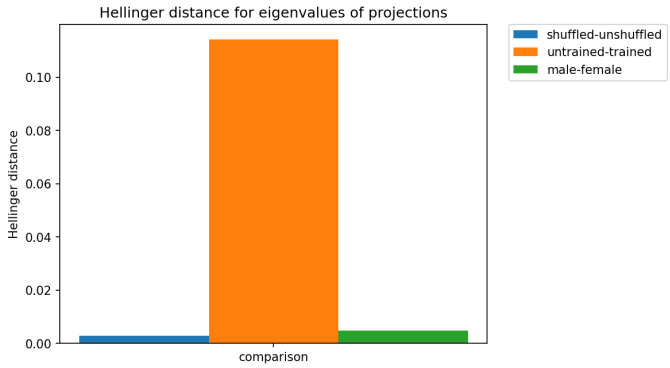
Figure A.5: Mean classification scores on the responses of multiple differently trained SSNs and the untrained SNN.



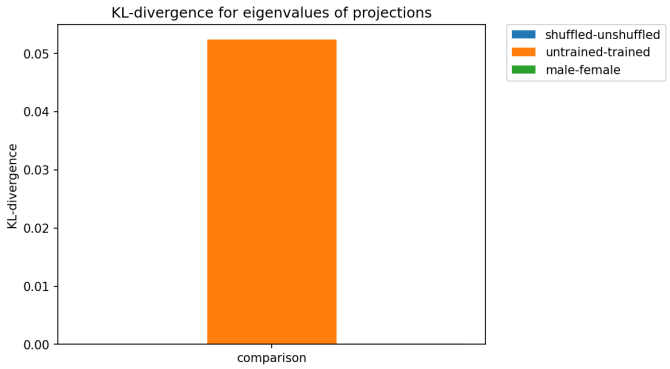
(a) Hellinger distance between weight distributions of different SNNs.



(b) KL-divergences between weight distributions of different SNNs.



(c) Hellinger distance between eigenvalue distributions of different SNN responses.



(d) KL-divergences between eigenvalue distributions of different SNN responses.

Figure A.6: Comparison of different SNNs their weight and eigenvalue distributions.

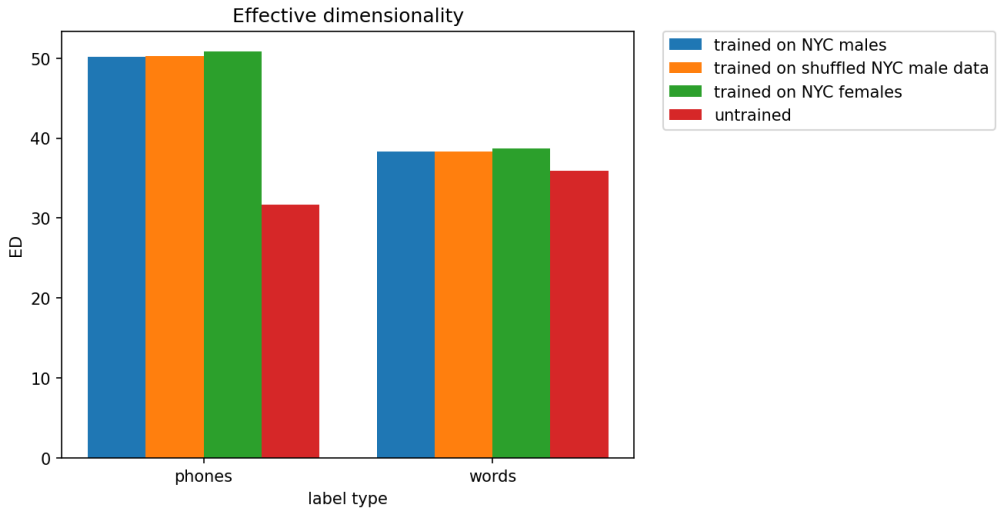
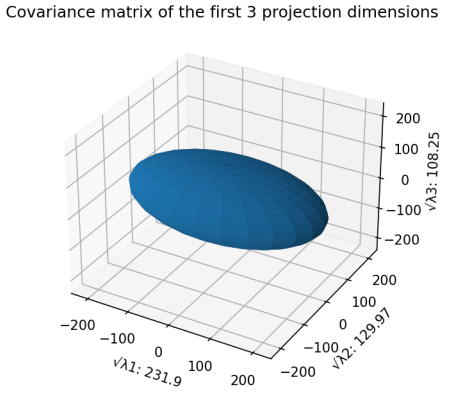
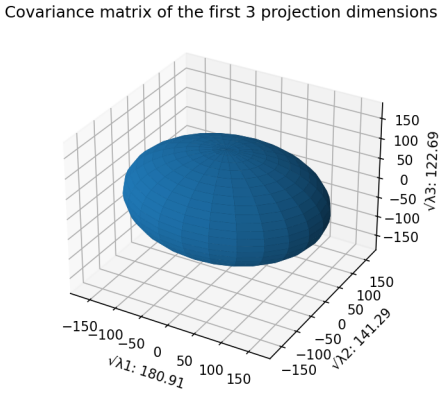


Figure A.7: Effective dimensionality of the projections (responses) of multiple trained SNNs and the untrained SNN on the same data.



(a) Untrained SNN responses covariance matrix of first 3 PCs



(b) Trained SNN responses covariance matrix of first 3 PCs

Figure A.8: Covariance matrices of the first 3 PCs after PCA projection of the responses of an untrained and trained SNN to phones, uttered by NYC males.