What's Natural About Natural Classes?
Towards a Computational, Historically-Informed Distinctive Feature Theory

Valerio Pepe (valeriopepe@college.harvard.edu)
Harvard SEAS

Abstract

Distinctive feature theory in phonology is notorious for not representing non-English languages accurately, and for having too many features to be cognitively plausible, due to being arbitrarily chosen by Chomsky and Halle in the 1960s (Chomsky & Halle, 1968). In this paper, we attempt to rectify this by proposing a sparse autoencoder-driven set of distinctive feature, through a graph representation of the 50 most common phonemes across languages (Cunningham, Ewart, Riggs, Huben, & Sharkey, 2023). We bring the number of features used down from 28 to 15 while retaining important structures within human language, and find that some featutes recognized by phonologists are empirically present in real-world data, while others are likely artificially manufactured to better describe English. However, our set cannot represent natural classes as richly, and we suggest using it for historical analysis of language more than analysis of contemporary language.

Introduction

Phonology is the branch of linguistics concerned with the study of the sounds of language, and how they change across time (diachronic phonology) or across words (synchronic phonology). As an example, imagine pronouncing the words 'cut' and 'cutter'; American speakers will pronounce the first with a hard 't' sound at the end, but will not do the same with the 't' in the second, pronouncing it like something closer to a 'd' or 'r'. This is called *intervocalic tapping*, and is one of the most common sound changes in synchronic English, capturing this change in pronounciation of both 't' and 'd' sounds when placed in between vowels (Ladefoged, 1993). It's not difficult to accept that 't' and 'd' may be considered similar sounds by phonologists, but why exactly are they similar? What do they have in common that makes them more similar to each other than to 'a'?

To answer this question, we can use distinctive features: distinctive feature theory represents each sound (we'll use the technical term 'phoneme' from here on) as a unique vector of binary features representing where and how the sound is produced in the human vocal tract. We can then keep track of how these features change in specific environments, and call those our 'rules'. As an example, see the features of the phonemes for /t/[1] and /d/ (we only write down the features present in each phoneme for clarity, ignoring the ≈ 20 that are not):

$$\text{/t/} = \begin{bmatrix} \text{+consonantal} \\ \text{+coronal} \\ \text{+anterior} \end{bmatrix} \quad \text{/d/} = \begin{bmatrix} \text{+consonantal} \\ \text{+voice} \\ \text{+coronal} \\ \text{+anterior} \end{bmatrix}$$

We can therefore group up both /t/ and /d/ in one class by selecting all phonemes that contain [+consonantal, +coronal, +anterior] (Chomsky & Halle, 1968). Note that since we did not specify a [+voice] constraint, we are picking up both phonemes with [+voice] and without it, so we are picking up both /t/ and /d/. Additionally, note that this only includes /t/ and /d/ by construction, since these features were constructed to follow common sound patterns in English (i.e. there is no other phoneme that is +consonantal, +coronal and +anterior, so our group will only contain these two phonemes). Such collections of phonemes are called *natural classes*; here, /t/ and /d/ form the natural class [+consonantal, +coronal, +anterior].

Noting that the 'tap' sound that these are pronounced as has the IPA character /ɾ/, we can write our intervocalic tapping rule as:

$$\begin{bmatrix} \text{+consonantal} \\ \text{+coronal} \\ \text{+anterior} \end{bmatrix} \rightarrow /ɾ/$$

.

As can be seen above, these features work well for simple, frequent English phonological rules, since the system was designed around them post-hoc. However, this is not true for all languages, and distinctive feature theory frequently has to be amended to make the features work in other languages. A famous example of this can be found in Geert Booij's 'Phonology of Dutch', where the author invents the feature [+French] to describe phonemes loaned from French, since they are granted specific exemptions from standard Dutch phonology (Booij, 1999). Additionally, the sheer number of features (28) makes the standard set of features utilized also cognitively implausible: is

[1] We make use of International Phonetic Alphabet (IPA) notation in this paper: // around a symbol denotes the sound that symbol represents in the IPA, e.g. /k/ is the sound that 'car' begins with. We do not assume familiarity past this, and will explain the relevant parts when needed.

phonology really a space with 28 degrees of freedom, especially when so many features are mutually exclusive with others, as well as given the constraints on human speech production due to human anatomy (Lisker & Abramson, 1971)?

The goal of this project is therefore to come up with a feature set that is more likely to work for languages that are not necessarily English, and is more cognitively plausible. For the purposes of this project, we will determine a set to be more cognitively plausible if it has fewer features (we want to cut them down by at least 50%), and has simpler contrasts, however we may define them. We would also like the features to still be able to describe common sound shifts, and, ideally, we would derive these new features with no influence from the standard feature set at all, in order to ensure the least amount of bias possible.

To achieve this, we propose a sparse-autoencoder-based system, which utilizes historical sound shift data represented as a graph as its ground truth and through which we will attempt to bottleneck the number of features, which we will then interpret to extract our desired feature set

Related Work

Linguistics

We will first give some background on the linguistic aspects of this paper: if one is already familiar with phonology, this part can be skipped.

Phonology As mentioned above, phonology is the study of the sounds used in human language. For this paper, the relevant portions are the technical names of the places and modes of articulation that phonologists utilize, as well as some International Phonetic Alphabet (IPA) symbols.

The places of articulation necessary here are only the most coarse-grained: labial, coronal, and dorsal. A 'labial' sound is one produced with the lips, a 'coronal' sound is one produced with the front of the tongue, and a 'dorsal' sound is one produced with the back of the tongue. Examples of labial sounds are /b/ or /p/, examples of coronal sounds are /t/ and /d/, and examples of dorsal sounds are /k/ and /g/. Notice how in pronouncing them one's tongue moves further and further back: these three areas roughly correspond to sounds produced at the front of the mouth, in the middle of the mouth, and at the back of the mouth (Hayes, 2008).

The modes of articulation that we will use here are plosives, liquids, nasals, fricatives, and vowels. 'Plosives' are made by closing the mouth completely (such as the sounds listed above) and have self-explanatory Latin characters as their IPA symbols (except /ʔ/, the glottal stop, the sound denoted by the hyphen in 'uh-oh!'). 'Liquids' are made by moving sound around the sides of the tongue: these are mostly /l/, /r/ and their derivatives, which, in the IPA, tend to have a lot of tails, e.g. /ɭ/, the retroflex lateral approximant, which is like an /l/ sound but further back in the mouth, or /ʁ/, the r-sound in French. 'Nasals' are sounds made with the use of the nose: think /n/, /m/, and -ng sounds, denoted by /ŋ/. 'Fricatives' are sounds that use turbulence, like /f/ or /v/, and are divided into sibilant and non-sibilant, with sibilant sounds being sounds like /s/ and /z/. There are more fricatives than any other mode of articulation on the IPA, so there are a lot of unconventional symbols used here: /θ/ and /ð/ for 'th', /ɣ/, /χ/, and /Φ/ and /β/ for various 'ph' or 'gh' sounds. Finally, vowels; these are self-explanatory, they are sounds made with the mouth completely open, and are sounds like /u/, /a/, /o/, /y/, /ə/ (an 'uh' sound), or /ʊ/ (an 'oo' sound) (Hayes, 2008).

Distinctive Feature Theory Distinctive feature theory is a theory of phonology developed by Noam Chomsky and Morris Halle in 1968 in their book 'The Sound Pattern of English' (henceforth SPE) (Chomsky & Halle, 1968). It breaks down any human sound into a vector of 28 binary features, which we will briefly explain.

First come the manner features, [consonantal], [sonorant], [continuant], [delayed release], [approximant], [tap], [trill], and [nasal]. These are consonant-only features, used to split up modes of articulation: most consonants are [+consonantal] (except /h/), and sonorant sounds are sounds that are not plosives (i.e. liquids, nasals and fricatives), continuants are sounds that do not ever close any part of the mouth completely (i.e. anything but plosives and nasals, since nasals close the velum, a part of the mouth that connects to the nose). [delayed release] is a marker for most fricatives, [approximant] is all fricatives and some other sounds where the lips come together but do not close, [tap] and [trill] are used for specific /t/-like and /r/-like sounds, and [nasal] is a marker for all nasal sounds (Chomsky & Halle, 1968).

Then the laryngeal features: [voice] marks if the sound is produced by vibrating the vocal chords, i.e. /f/ is [-voice] but /v/ is [+voice] (one can feel this by holding a hand to their throat as they say these and feel the vibrations), and [spread glottis] and [constricted glottis] are used for very specific, non-standard sounds like implosives (which do not exist in American English) or the glottal stop /ʔ/.

Finally, the 14 place features. 5 of these are mainly used in vowels (high, low, front, back, tense) and describe where the tongue is in the mouth as the sound is produced, and the others

rely on features like [labial], which, if present, divided into [round] (rounded lips) and [labiodental]. Then, [coronal] divides into [anterior] (which uses the front of the mouth, the palate's ridges), [distributed] (uses the side of the tongue), and [strident] (loud fricatives, /s/, /z/ et similia). Finally, we have [lateral], another feature for airflow around the sides of the tongue, and [dorsal].

Distinctive features as described in SPE (and most subsequent works) have been criticized on account of being cognitively implausible given their high number, as well as their anglocentric bias (most languages do not distinguish between /t/ and /θ/, for example) (Mielke, 2004).

Computer Science

(Sparse) Autoencoders An autoencoder is a neural network architecture where the output's goal is to reconstruct the input. To do this, the hidden layers are intentionally more narrow (i.e. have fewer neurons) than the input and output layers: this 'bottleneck', as it is known, forces the network to learn a meaningful representation of the input to be able to reconstruct it after the bottleneck (Kramer, 1991).

Sparse autoencoders are a type of autoencoder that enforces a sparsity constraint on the bottleneck layer: this can be thought of almost as a regularization on the activations of a layer in the middle of the network rather than one at the output. These were most recently and famously the subject of **cunningham 2023**, where it was shown that enforcing the sparsity constraint, if well-calibrated, gives more intepretable features than non-sparse alternatives.

Graph Representation Learning Broadly speaking, graph representation learning is a type of statistical learning where the input is represented as a graph's vertex or neighborhood, and the learner's objective is to learn the main features of the neighborhood in order to further some downstream task (in our case, dimensionality reduction of the input data) (Hamilton, 2020).

In our case, our network will be fed a vertex (the connections from one phoneme to all the other phonemes it patterns with, phonetically) and will have to infer how those connections can be compressed into a smaller number whilst retaining some sparsity in the final representation.

Dataset The dataset containing the sound shifts is Index Diachronica, a database made to catalog historical language data for those wishing to construct fictional languages as a hobby. Notwithstanding the unacademic nature of the database, though, the sound shifts are meticulously referenced and rooted in academic texts, so I have no reason to doubt the data. I just downloaded their integral .csv from the website and used that data, containing 13000 sound shifts in total across a variety of languages and across time.

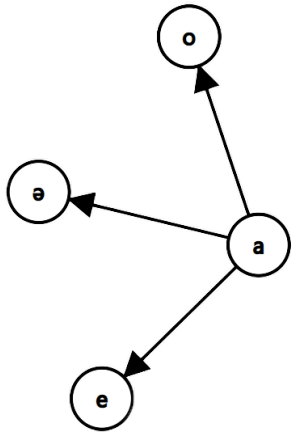

Figure 1: A detail of the phoneme graph, the neighborhood of the phoneme /a/

Methods

Graph Construction We went through Index Diachronica's database of sound changes: we will make reference to these by dividing sound shifts into a 'starting phoneme' and an 'ending phoneme', meaning a phoneme before and after it undergoes the sound shift. Thus, for every starting phoneme, we kept track of all of its ending phonemes across all the sound shifts it was a part of, and counted how many times each one appeared. We then built our graph considering the starting phonemes as our vertices, and connected a phoneme x with some other phoneme y if y was x's ending phoneme in at least 10% of the sound shifts x was a part of. Edges were unweighted: we made no distinction between the phoneme having a frequency of barely over 10% or being the only phoneme the starting phoneme turned into[2].

Since the features will be created directly from the sound shift data, low-quality or underdocumented phoneme data could cause the autoencoder to settle on an awkward representation and inject noise into the features it settles on as the most representative of each phoneme, so it had to be removed. To make sure every phoneme used to extract features was well-represented in the sound shift data, we therefore only kept the top 50 most common phonemes represented in human language as detailed in (Moran & McCloy,

[2]This was mainly an engineering consideration since the autoencoder stopped converging when we tried to factor in relative frequency, so this is something future work can explore.

2019) (a reputable database for the frequency of linguistic features across languages) that appeared at least 15 times in Index Diachronica (the full list of phonemes kept can be found in the appendix).

Autoencoder Details We used a custom autoencoder architecture, which we intentionally kept to three layers: the input layer, the bottleneck, and the output layer. This choice was made to promote interpretability in the network; papers like (Bricken et al., 2023) and (Cunningham et al., 2023) suggest that, together with a sparsity constraint on the autoencoder, shallow networks make it easier for more interpretable features to arise on account of the simpler non-linear relationships that we allow (we can think of this almost as an intentional constraining of the expressive power of the network), so arranging our network like this makes it easier for us to be able to see specifically what the features will be during the inference part of the project.

The autoencoder's bottleneck's width was varied from 10 to 20 in preliminary experiments, but we found that a width of 15 (i.e. the use of 15 features by the autoencoder to describe the phonemes) provided the best results, with some important features not appearing as cleanly in the results when the width too small, or an excessive number of repetitive features when the width was too large.

We used sigmoid activation functions to go from the input layer to the bottleneck layer and from the bottleneck to the output layer; this clips the activations between 0 and 1 and had the added benefit of making the interpretable as probabilities of a certain feature being activated.

The goal of the autoencoder is to take in a phoneme, represented by its connections to the other phonemes, pass it through the bottleneck to reduce the dimensionality of its very sparse connections, and then reconstruct the vertex on the other side of the autoencoder. We therefore have only 50 pieces of input data (one for each phoneme), and one epoch will thus be defined as one pass through all 50 phonemes.

The autoencoder was trained with an AdamW optimizer (Loshchilov & Hutter, 2019) with the default PyTorch parameters (a learning rate of 10^{-3}, weight decay of 0.01) – changing these parameters was tried in preliminary experiments, but it did not lead to improvements, so the default parameters were kept. We also trained it for 4000 epochs or until it reached perfect reconstruction accuracy, whichever came first: this may seem excessive and like it would lead to overfitting, but given our small amount of input data and that neural network loss convergence usually takes thousands of data points even for very shallow networks, this type of sample complexity is reasonable.

Additionally, overfitting was not a point of concern for this project: these 50 phonemes have been stable for thousands of years, and barring some individual variation in their actual production from speaker to speaker, it is extraordinarily unlikely that new phonemes will arise that our model will not take into consideration, especially in the long term (new phonemes being used would have to mean a large-scale shift in the way humans produce language for them not to be captured within IPA symbols that already exist).

Loss Function The characteristics of the construction of the graph and the autoencoder as described above also constrain the loss function in usual ways. As is typical for classification tasks, our base loss function was an element-wise Cross-Entropy Loss applied to the autoencoder's output to measure its performance. This loss function was chosen because of the graph representation aspect of this project: an edge between vertices can either exist (1), or not exist (0), so we can frame this as a classification task between these two and train the network that way. Other loss functions, such as mean squared error or Kullback-Leiber Divergence were also tested, but they were more inaccurate than cross-entropy loss, as well as being badly defined in this context (this is not a regression task, an output being 1.1 or -1 are not well-defined in this context and should just be clipped to 1 or 0 anyways, making cross-entropy loss most useful).

We then apply two modifications to the cross-entropy loss, to make it fit for our purpose. First we apply a sparsity constraint as outlined in (Cunningham et al., 2023) as mentioned above, in order to further increase the interpretability of the network's features. We apply this constraint through an L_1 penalty to the autoencoder's middle layer, multiplied by some $\alpha = 0.001$. This value was tuned to be similar to the paper's $\alpha = 0.000864$, and empirically adjusted for the specific problem setting, by seeing that values above 0.001 removed too much structure from the underlying data, and values below it admitted too much noise into the features to make them easily interpretable.

Second, since we specified an edge exists between two phonemes in the graph to be learned if and only if one is an ending phoneme for the other at least 10% of the time, it easily follows that any phoneme can have at most 10 neighbors. Given there are 50 phonemes and we feed the adjacency matrix representation into the network as our input data, this leads to input data that is quite sparse. Initial tests, thus, had the model 'learn' to always output 0, since it would result in a small loss re-

gardless. To combat this, in calculating the loss, we don't calculate the loss between the input x and its reconstruction $\hat{x}$, but between the input x and $2.5\hat{x}+1$. This method ensures that every true 0 in the input gets updated to 1 in calculating the loss (and every 1 gets updated to 2.5, ensuring that the difference between a 1 and a 0 is still there, such that we don't end up with outputs that are just a vector of 50 ones), increasing the penalty for always predicting 0 and thus coaxing the model into outputting numbers that are not 0 and learning a useful representation of the data.

Inference Given the sigmoid activations that characterize the bottleneck layer, the standard interpretation of the features would be as probabilities: if for some input a feature is more than 0.5, then we round that to 1 and say it is 'active', and if it is less than 0.5 we round it to 0 and say it is 'infactive'. However, this relies on the assumption that the activations are distributed along the entire $[0,1]$ interval, so there is a clean separation at 0.5 about what is and is not part of a feature. This assumption was not true in this data: we found features where the separation is much more natural if centered at a threshold that is not 0.5, where setting the threshold at 0.5 actually takes away some nuance in the data, and adds noise that can be avoided. As an example, in Fig. 2, we can see that Feature 9 in the results clearly has two clusters with a separation at 0.7, not 0.5.

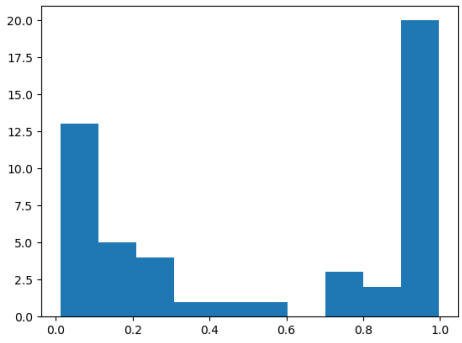

Figure 2: A histogram of the activations of Feature 9 over the 50 input phonemes

To separate these features more cleanly, therefore, we introduce a Gaussian Mixture Model (GMM) to cluster the data into two clusters[3]. We then separate each datapoint into an 'inactive' or 'active' feature depending on the cluster that it was assigned to during the learning of the mixture model.

Results

Figure 3 showcases the best results obtained in the experiment, the ones we will be discussing at length. Results from other experiments, like other sparsity levels and a 10- and 12-wide autoencoder, will be included in the appendix and discussed in the discussion section of this paper, but to a lesser extent. All autoencoders discussed reached perfect reconstruction accuracy, meaning we can be sure they learned some meaningful structure in the data.

From the results of Figure 3 (Panel (c)'s results, specifically), we can extract the following features. We named them pre-emptively to not have to repeat them once more in naming them, these names will be discussed in the 'Discussion' section[4]:

Front (Labial/Coronal): ['a', 'i', 'e', 'o', 'ɛ', 'ə', 'ɪ', 'ɑ', 'æ', 'y', 'g', 'p', 'b', 't', 'd', 'm', 'β', 'f', 'v', 's', 'z', 'ʒ', 'ʃ', 'w']

Middle (Coronal/Dorsal): ['ʊ', 'g', 't', 'd', 'ʈ', 'c', 'n', 'ŋ', 'ɲ', 'ɸ', 'ð', 's', 'z', 'ʒ', 'ʃ', 'x', 'ɾ', 'l']

Plosives, Fricatives: ['ɛ', 'æ', 'q', 'g', 'p', 't', 'd', 'c', 'ɳ', 'ɸ', 'ð', 'f', 's', 'z', 'ʒ', 'ʃ', 'x', 'ʁ', 'ʔ', 'h']

Non-i vowels, liquids, fricatives, voiced stops: ['u', 'a', 'o', 'ɔ', 'ə', 'ʊ', 'y', 'q', 'g', 'β', 'ð', 'χ', 'f', 'v', 'z', 'ʁ', 'r', 'ɾ', 'ɭ', 'w', 'l', 'h']

Every Vowel, Back consonants: ['u', 'a', 'i', 'e', 'o', 'ɛ', 'ɔ', 'ə', 'ɪ', 'ʊ', 'ɑ', 'æ', 'ɯ', 'y', 'c', 'ɲ', 'ɳ', 'χ', 'ʁ', 'j', 'ʈ']

Nasals, Fricatives: ['q', 'm', 'n', 'ŋ', 'ɳ', 'β', 'χ', 'f', 'v', 's', 'ʒ', 'x', 'ɣ', 'ʁ', 'ɾ', 'ʔ', 'w', 'h']

Front vowels, Non-Plosives: ['a', 'i', 'e', 'ɛ', 'ə', 'ɪ', 'æ', 'q', 'z', 'x', 'ɣ', 'ɾ', 'ʈ', 'ʔ', 'h']

Stops, Non-sibilant fricatives: ['ɪ', 'ɑ', 'ɯ', 'q', 'b', 't', 'd', 'ʈ', 'c', 'ɸ', 'ð', 'χ', 'v', 'ɣ', 'ʁ', 'ɭ']

Non-mid Vowels, Fricatives: ['u', 'o', 'ɔ', 'ʊ', 'ɑ', 'y', 'p', 'b', 'm', 'ɳ', 'β', 'ɸ', 'f', 'v', 's', 'ʒ', 'ʃ', 'x', 'ɣ', 'ɭ', 'w']

Nasals, front vowels, non-front consonants: ['ɛ', 'ɪ', 'ɯ', 't', 'd', 'ʈ', 'c', 'n', 'ŋ', 'ɲ', 'ɳ', 'ð', 'z', 'ʁ', 'j', 'r', 'ɾ', 'ʈ', 'ɭ', 'l']

Front Vowels, Non-mid consonants, liquids: ['e', 'ɛ', 'ɪ', 'ɯ', 'y', 'p', 'b', 'ʈ', 'm', 'ŋ', 'β', 'ɸ', 'f', 'r', 'ɾ', 'ʈ', 'ɭ', 'ʔ', 'l', 'h']

[3]There are rare cases of feature activation histograms that look tri-modal more than bimodal, but these do not show up in the final results set, so we decided to make the mixture model assume all data was bimodal.

[4]Note: Most of these characters are not supported in default LaTeX, so I had to manually import them. Apologies if any are missing.

Back vowels, non-front consonants: [’o’, ’ɔ’, ’ʊ’, ’ɑ’, ’ʈ’, ’c’, ’s’, ’ʒ’, ’ʃ’, ’x’, ’ɾ’, ’l’]
All vowels, mid consonants: [’u’, ’a’, ’i’, ’e’, ’o’, ’ɛ’, ’ɔ’, ’ə’, ’ɪ’, ’ʊ’, ’ɯ’, ’y’, ’g’, ’p’, ’t’, ’d’, ’ʈ’, ’ɲ’, ’ɳ’, ’β’, ’ð’, ’ʒ’, ’ʈ’]
Non-Middle Vowels/Plosives: [’u’, ’a’, ’e’, ’o’, ’ɛ’, ’ɔ’, ’ə’, ’ʊ’, ’ɑ’, ’æ’, ’ɯ’, ’y’, ’q’, ’k’, ’g’, ’b’, ’m’, ’n’, ’ŋ’, ’χ’, ’ɣ’, ’ʔ’]
Front Vowels, Liquids and Nasals: [’a’, ’i’, ’e’, ’ə’, ’ɪ’, ’ɑ’, ’æ’, ’p’, ’b’, ’c’, ’m’, ’n’, ’ŋ’, ’ɲ’, ’ɸ’, ’f’, ’v’, ’j’, ’ʔ’, ’w’, ’l’, ’h’]

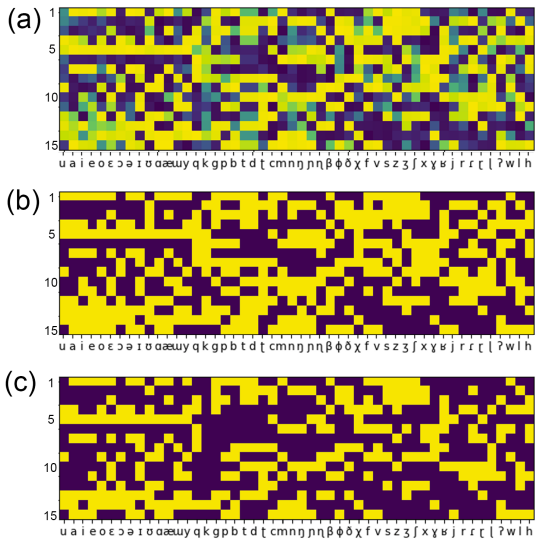

Figure 3: The raw feature scores extracted from the autoencoder. Panel (a): Raw scores, Panel (b): Rounded scores (0.5 threshold), Panel (c): GMM-rounded scores. Color code: yellow indicates a score ≈ 1, purple indicates a score ≈ 0.

Discussion

Now that we have a coherent set of features, we can discuss the results obtained, how much we achieved our goal of finding a better feature set for distinctive feature theory, and what these features may mean for the way humans perceive phonemes.

Firstly, the way the features were named. Notice that, differently from the original SPE features, where each feature is cleanly described by a single characteristic (even though it may not be an obvious one), here most features contain a mixture of more than one characteristic, with only the first two features, ‘Front’ and ‘Middle’, being as elegant as the SPE features. All of the others are a mixture of different modalities of sound production, such as Feature 4, ‘Non-i vowels, liquids, fricatives, voiced stops’, or different locations in the mouth, such as ‘Front Vowels and Nasals’, Feature 15.

These messy names imply that the coding derived by the autoencoder is completely different from the SPE coding. If we derive features solely from historical data, thus, this suggests that SPE is very different from the trends available in the actual data, and is a very heavily finetuned set of features made to be as elegant as possible for English.

For instance, notice that all the vowel-exclusive location features are completely absent: consonants and vowels both share three characteristics based on where they are produced in the mouth, ‘front’, ‘mid’, and ‘back’ and their combinations (like ‘front’ and ‘back’ joining to make ‘non-mid’, etc.). Additionally, [+voice] is not a clear-cut characteristic: we don’t have a feature that clearly separates the unvoiced /p/, /t/, /k/, /f/ etc. from the voiced /b/, /t/, /g/ and /v/, respectively. This is quite odd: voice is usually considered one of the simplest and most ubiquitous features given it has a very clear physical trigger (the vibration of the vocal cords). Evidently, though, this suggests that when we compare it to historical data, voicing was not as widespread across languages as it is today. In fact, this matches up with historical linguistic intuition: Classical Latin lacked a distinction between /f/ and /v/ or /s/ and /z/, and proto-Greek had no fricatives that were not /s/. Thus, we believe it is reasonable to chalk up this difference to historical data, but this difference between the common characteristics of language today and language historically leads us to posit that the feature set recovered by the autoencoder may be more useful as a tool for historical analysis more than contemporary language.

This idea that the feature set recovered may be more useful for historical analysis is further reinforced by collisions in feature space: even though the autoencoder itself is able to have perfect reconstruction accuracy for the features, once we apply the GMM we get phonemes that look exactly the same. As an example, /t/ and /d/, in this set of results, share exactly the same features, and are not distinguishable. This is an issue, but one that can be fixed by small manual edits of the features, such as adding either one of them to the ‘Non-mid Vowels, Fricatives’ feature, where it would fit, and therefore find a difference that way. However, historically, /t/ and /d/ pattern together incredibly often, so it is not surprising that, once we take a more nuanced look at the feature space, the two are literally indistinguishable, especially since there exist languages today that do not distinguish them (Passamaquoddy and Hawaiian in

North America, for instance) (Leavitt, 1996). Another quirk is the placement of the phoneme /k/, which is only present in Feature 14, 'Non-Middle Vowels/Plosives' and nowhere else, which is quite awkward (/k/ is not just a non-middle plosive, it shuld also b a back plosive, a back consonant, etc.).

This being said, there are also significant similarities between the SPE set of features and ours. Both have a very clear distinction between consonants and vowels: Feature 5, 'Every Vowel, Back consonants', includes a very clean 'vowel' feature that includes all the vowels in the training data, for example. This suggests that some features found in SPE, and the way linguists treat language, *are* universal to human languages across time, and it is correct to include them in a feature set that supposedly holds features to describe every single human phoneme possible. There also seems to be a set of distinctions that traces the 'sonority hierarchy', a phonological concept dividing phonemes by how open the mouth is while producing them, going from vowels (most open), to liquids, to nasals, to fricatives, to stops (most closed) (Everett, 1996). In the feature set recovered, phonemes seem to appear in the same feature only if they are close to each other in this hierarchy: there is a 'Nasals, Fricatives' feature, a 'Front Vowels, Liquids and Nasals' feature, a 'Stops and Non-sibilant fricatives' feature, but not a 'Vowels and Stops' feature, or a 'Liquids and Fricatives' feature. Phonemes placed differently in the sonority hierarchy are thus commonly placed in different features, indicating that there is some natural contrast between them.

In terms of what this set predicts about human phonetics across time and across languages, we can say that the set we have recovered suggests there should be a much greater emphasis on the place of articulation for phonemes, since almost every feature includes more than one mode of articulation, but only one place of articulation (or one for vowels and one for consonants). Additionally, while consonants and vowels should be seen as very distinct sets of phonemes, each patterning mostly with phonemes within the same set, vowel- or consonant-exclusive features can be avoided, since the two sets can share features and still find a way to be distinct from another (the 'Every Vowel, Back consonant' feature and the 'All vowels, mid consonant' feature together include all the vowels but no consonants).

We also have interesting results from a computer science perspective, as we are clearly seeing superposition within the features (Elhage et al., 2022). Almost every feature exhibits a mixture of more than one uncorrelated characteristics: the sonority-hierarchy-adjacent features from earlier prove this (no phoneme will be both a vowel and a fricative, for instance), but we can also note that there are many features that just specify some vowels united by a place of articulation, and some consonants united by a place of articulation e.g. 'Front vowels, Non-mid consonants, liquids', or 'Back vowels, non-front consonants'. Since there is no overlap between these, we can argue that they represent distinct directions in feature space that are being superimposed onto one another, exactly what we would expect in neural network superposition. The sparsity constraint also greatly affects the results: Fig 5 and Fig 4 in the appendix show the effects that changing the sparsity parameter α has on the feature activation space, showing that increasing α gives us clearer-cut but less expressive (almost useless) features and decreasing it gives us less-interpretable but more expressive (and thus less sparse) features, as expected.

In conclusion, we have used a sparse autoencoder-based approach to leverage historical sound shift data and derive a new, computationally- and historically-informed set of distinctive features to use in phonology. These features show that some common contrasts in phonology are backed up by historical data – vowels vs. consonants, the sonority hierarchy, the distinction between places and modes of articulation – but others, such as many vowel- or consonant-specific location features, some finegrained manner features like trills and taps (which all pattern together with liquids at large anyway), and all the laryngeal features (including [+voice]) are not, and are mostly artifacts of feature sets useful for specific languages. However, this universality of our feature set is also its downfall: 'it is a jack of all trades and master of none', meaning that it has sacrificed language-specific abilities (like the distinction between /t/ and /d/ in modern speech) for the ability to capture higher-level, more abstract shifts in sounds, rendering it a more efficient coding space-wise (15 features instead of 28) and one that more effectively captures how humans perceive sounds, but also rendering it very inefficient from a phonological standpoint, as it cannot capture common modern natural classes.

Further work may try to achieve this last desideratum by biasing the training data towards modern languages instead of considering every language, by somehow setting a minimum number of features that each phoneme must abide by to avoid settings like /k/'s, or by incorporating the difficulty of reproducing common sound shifts into the loss function to emphasize the usability of the feature set.

Appendix

Full Set of Phonemes

The full set of phonemes used is as follows:

Vowels: ’u’,’a’,’i’,"e","o","ɛ",’ɔ’,"ə","ɪ","ʊ","ɑ","æ","ɯ","y"

Plosives: "q",’k’,"g",’p’,"b","t","d","ɖ","c",’ʔ’

Nasals: ’m’,"n",’ŋ’,"ɲ","ɳ"

Fricatives: "β","ɸ","ð","χ","f","v","s","z",’ʒ’,’ʃ’,"x",’ɣ’

Liquids: "ʁ","r","ɾ","ɽ","ɭ","l"

Other: ’j’,’w’,"h" (/h/ is a non-consonantal consonant; /j/ and /w/ are what is known as a ‘glide’, sounds that have more than one place of articulation)

Non-Sparse and Over-Sparse Results

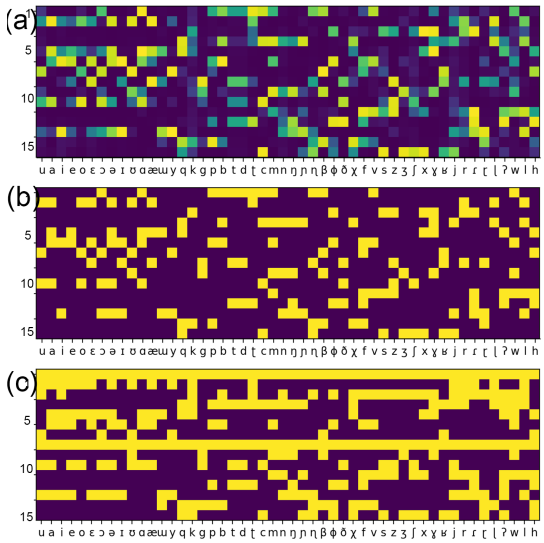

Figure 4: Autoencoder results with $\alpha = 0.002$ (‘Oversparse’, panels same as Figure 3)

Doubling α to 0.002 leads to clearly unusable features: notice that Features 1 and 7 are active for all phonemes, and there are very strange patterns in the others, such as feature one being on for half the vowels, off for the other half, and the on on for the liquids, or /ð/ just being composed of features 10 and 12.

Setting $\alpha = 0$ makes the feature activations a lot less sparse even just by inspection, and reinforces a lot of the similarity between similar phonemes: vowels now have a cluster of features that seem to

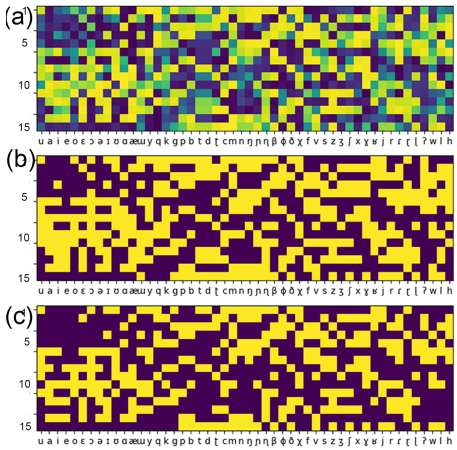

Figure 5: Autoencoder results with $\alpha = 0$ (‘Undersparse’, panels same as Figure 3)

be mostly concentrated around them, not just a single one, and most plosives seem to pattern together. However, the lack of sparsity invites a lot of noise: the vowel-dominated features are also present in a lot of the consonants, and there is no single, clean feature in which all the vowels are present, which is a huge loss for interpretability.

These results are likely due to the fact that there are $2^{15} = 32768$ possible combinations of features and only 50 phonemes, so this is a very overdetermined space, and without the right amount of sparsity we either underconstrain it, leading to situations like $\alpha = 0$, or we overconstrain it and we obtain the oversparse $\alpha = 0.002$ graph, even if both can obtain perfect reconstruction accuracy since they can essentially one-hot encode phonemes given the huge feature space.

In light of these two results, it thus makes sense to go somewhere in the middle, around $\alpha = 0.001$, where the sparsity seems to be about right and leads to more meaningful features.

Data Availability

All the code, data, and models for this project can be found at https://github.com/vpepe/Neuro-240-Project. I wrote all of the code myself using PyTorch.

References

Booij, G. (1999). *The phonology of dutch*. Oxford, England: Clarendon Press.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., ... Olah, C. (2023). Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. (https://transformer-circuits.pub/2023/monosemantic-features/index.html)

Chomsky, N., & Halle, M. (1968). *The sound pattern of english*. London, England: MIT Press.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., & Sharkey, L. (2023). *Sparse autoencoders find highly interpretable features in language models.*

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., ... Olah, C. (2022). *Toy models of superposition.*

Everett, D. L. (1996). Donald a. burquest david l. payne (1993). phonological analysis: a functional approach. dallas: Summer institute of linguistics. pp. viii+179. *Phonology*, *13*(2), 269–274. doi: 10.1017/S0952675700002128

Hamilton, W. M. (2020). *Graph representation learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning.

Hayes, B. (2008). *Introductory phonology*. Chichester, England: Wiley-Blackwell.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *Aiche Journal*, *37*, 233-243. Retrieved from https://api.semanticscholar.org/CorpusID:15907287

Ladefoged, P. (1993). *A course in phonetics* (3rd ed.). Florence, AL: Heinle & Heinle.

Leavitt, R. M. (1996). *Passamoquoddy-maliseet*. Munich, Germany: LINCOM.

Lisker, L., & Abramson, A. S. (1971, December). Distinctive features and laryngeal control. *Language*, *47*(4), 767. Retrieved from http://dx.doi.org/10.2307/412155 doi: 10.2307/412155

Loshchilov, I., & Hutter, F. (2019). *Decoupled weight decay regularization.*

Mielke, J. (2004). *The emergence of distinctive features*. Oxford University Press.

Moran, S., & McCloy, D. (Eds.). (2019). *Phoible 2.0*. Jena: Max Planck Institute for the Science of Human History. Retrieved from https://phoible.org/