Vowel formants predict place of articulation of following consonant in Kannada: Evidence for the autosegmental theory of phonology

Praveen Pallegar
pallegar@stanford.edu

Abstract

The autosegmental theory of phonology claims, among other ideas, that individual phonemes carry information about both preceding and following environments. In this study, I considered vowels preceding retroflex and dental consonants in Kannada. Using resonant frequencies (formants) of these vowels, I created a classifier to predict the place of articulation of the following consonant. My results suggest that phonemes do not exhibit a one-to-one correspondence with their higher-level representation, but rather that they carry information about their environment that humans can use as predictive capacity in the phonetic level of language comprehension.

Introduction

In the field of human language comprehension, we see an enormous computational task in the ability of humans to perceive, parse, and comprehend large amounts of information in real-time. On the semantic and syntactic levels of analysis, a great deal of research has shown that humans lighten this computational load by virtue of semantic/syntactic frameworks and association networks that allow us to leverage the predictive capacity of language to our advantage [2,6]. However, relatively less research has been done on the predictive capacity of phonemes, the smallest distinguishable unit of sound.

A landmark paper in phonology proposed the autosegmental model for phonology, claiming, among other ideas, that feature parsing of phonemes is not a bijective function, and that each phoneme additionally carries information about its environment [4]. Thus, such secondary information could be used to simultaneously predict phonetic environments, thereby further reducing the computational load of phonetic comprehension. More recently, Gow et al. showed that listeners could differentiate between segments that had merged completely through place assimilation. The authors hypothesized that the preceding vowel retained information about the underlying forms, allowing listeners to disambiguate meaning [3].

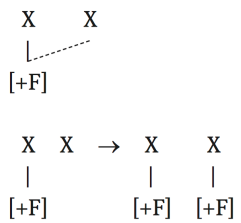

Fig 1. General assimilation paradigm in a(n) (top) autosegmental and (bottom) feature copying representation [5]

In the present study, I seek to look closer into this phenomenon using Kannada as my model language. In the phonemic inventory of Kannada, most consonants come in two places of articulation: dental and retroflex. Using a variety of machine learning algorithms, namely logistic regression, Gaussian Discriminant Analysis (GDA), support vector machine (SVM), semi- and un-supervised Gaussian Mixture Model (GMM), and neural networks, I will create a model of this classification problem to output the predicted place of articulation given the formants of the preceding vowel.

Methods

Data Collection

Data for this project comes from two native speakers of Kannada. They were told to record waveforms of spontaneous speech that included various minimal pairs of interest. Each minimal pair included the same vowel followed either by the dental or retroflex version of the consonant.

ಅನ್ನ → [ʌnːə] *rice*

ಅಣ್ಣ → [ʌɳːə] *brother*

In the above minimal pair, the vowel of interest (bolded) precedes a dental consonant in the first example and a retroflex consonant in the second example.

Data Processing

Using Praat software, vowels of interest were isolated and formants 1-3 (F1, F2, and F3) were extracted as the average over the vowel utterance. F1, F2, and F3 were used as features for each training example, and the label 1 was given to vowels preceding retroflex consonants while the label 0 was given to vowels preceding dental consonants.

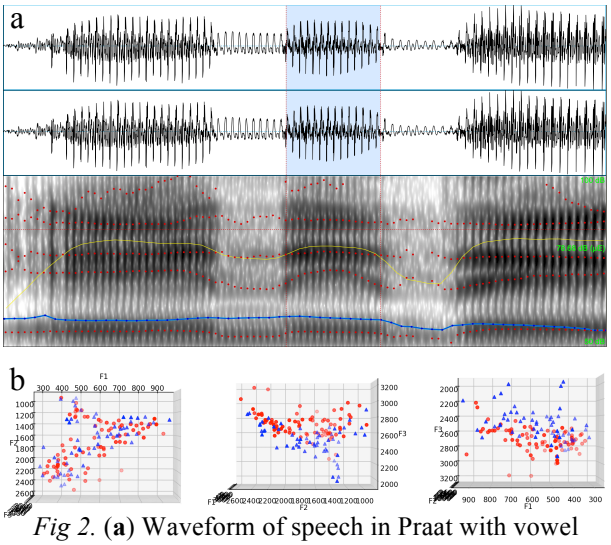

Fig 2. (**a**) Waveform of speech in Praat with vowel highlighted. (**b**) 3D representation of dataset along each set of axes (blue = 1, red = 0)

Logistic Regression

Using full-batch gradient descent, I created a classifier that updated parameters as per the Newton-Raphson method:

$$\theta := \theta - H^{-1}\nabla_\theta \ell(\theta)$$

where H^{-1} is the Hessian of the log-likelihood function $l(\theta)$, given by

$$\sum_{i=1}^{n} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))$$

and the prediction is represented by the sigmoid function of the inner product of the parameters and features.

Gaussian Discriminant Analysis

In the GDA model, we begin with the assumption that the class prior comes from a Bernoulli distribution whereas the features given the class are random variables distributed according to the multivariate normal distribution. Under these assumptions, we can use the Naïve Bayes' Assumption to simplify the log-likelihood function to the following:

$$\log \prod_{i=1}^{n} p(x^{(i)}|y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi)$$

Maximizing this provides the Maximum Likelihood Estimates of the parameters.

Support Vector Machine

Kernel methods in SVMs allow us to map the features into a higher-dimensional space and compute a decision boundary in this space before mapping back into the original dimension. In a high-dimensional feature space, the parameters θ can be implicitly represented as β with the following update rule:

$$\beta := \beta + \alpha(\vec{y} - K\beta)$$

Here, K represents the kernel corresponding to a feature map ϕ such that

$$K(x^{(i)}, x^{(j)}) \triangleq \langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$$

In this project, I compare linear, polynomial, Gaussian, and sigmoid kernels.

Gaussian Mixture Model

This model is an application of a broader algorithm known as the expectation-maximization (EM) algorithm. We use hidden random variables $z^{(i)} \sim Q_i$ where *i* denotes the *i*th example. In GMM we assume that each Q_i is a multivariate Gaussian distribution. In the expectation (E)-step, I calculate

$$w_j^{(i)} = Q_i(z^{(i)} = j) = P(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma)$$

for *j = 0* or *1* (the two labels in the dataset). Next, in the maximization (M)-step, I maximize the log-likelihood function given by

$$\sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \phi, \mu, \Sigma)}{Q_i(z^{(i)})}$$

with respect to each parameter to find the maximum likelihood estimate.

The above algorithm pertains to the unsupervised case, but in the semi-supervised case, we add the following term to our log-likelihood function:

$$\alpha \left(\sum_{i=1}^{\tilde{n}} \log p(\tilde{x}^{(i)}, \tilde{z}^{(i)}; \theta) \right)$$

where $\tilde{n}$ is the number of labeled examples and $\tilde{z}^{(i)}$ are visible labels such that

$$\ell_{\text{semi-sup}}(\theta) = \ell_{\text{unsup}}(\theta) + \alpha \ell_{\text{sup}}(\theta)$$

Neural Network

I created a neural network with 1 hidden layer using the sigmoid activation function for both the hidden and output layers. Using mini-batch gradient descent, I minimize the cost function given by

$$J_{\text{mb}} = \frac{1}{\text{B}} \sum_{i=1}^{\text{B}} \mathcal{L}^{(i)}$$

where

$$\mathcal{L}(\hat{y}, y) = -\Big[(1 - y) \log(1 - \hat{y}) + y \log \hat{y} \Big]$$

to derive updates for parameters *W* and *b*:

$$W^{[\ell]} = W^{[\ell]} - \alpha \frac{\partial \mathcal{L}}{\partial W^{[\ell]}}$$

$$b^{[\ell]} = b^{[\ell]} - \alpha \frac{\partial \mathcal{L}}{\partial b^{[\ell]}}$$

Testing and Feature Selection

For my implementation of logistic regression and GDA, I tested accuracy using k-fold cross-validation with k = 5. After randomly splitting the data into 5 partitions, I iterated through the algorithm 5 times, rotating which 4 partitions contributed to the training set and which partition was the testing set.

I additionally implemented backwards feature selection in logistic regression and GDA to determine the predictive capacity of individual features and optimize my model.

Results

Model	Test Accuracy	Train Accuracy
Log_Reg	0.7272	0.7627
GDA	0.7344	0.7696
SVM_sig	0.8148	0.7232
SVM_rbf	0.7037	0.7589
SVM_poly	0.5926	0.6339
SVM_lin	0.7778	0.7411
GMM_un	0.5294	--
GMM_semi	0.7143	--
NN	0.6149	0.7319

Table 1. Test and train accuracies for all models

Logistic Regression and GDA

For Logistic Regression, I assumed convergence when the update became smaller than 1e-5. I have no hyperparameters to report for GDA.

SVM

I used the SVM algorithm provided in the Scikit-learn library [7]. As we can see, the model with the highest accuracy was the SVM using a sigmoid kernel. Given the 3D representation of the data in Figure 1, a high-dimensional feature mapping would create a robust decision boundary.

GMM

I assumed convergence when the log-likelihood changed by less than 1e-15 per iteration. I selected this threshold to ensure convergence and thorough training. Additionally, I set the weight for labeled examples in the semi-supervised case to 20. The unsupervised model performed only slightly better than random guessing in almost all trials. However, the semi-supervised model achieved a higher accuracy. Below is a comparison of classifications from semi- and un-supervised GMM.

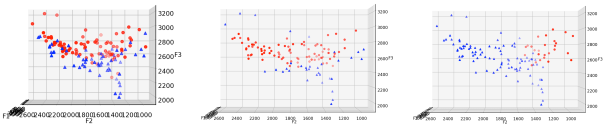

Fig 3. F3 vs F2 in (left) data, (middle) semi-supervised GMM, and (right) unsupervised GMM

Neural Network

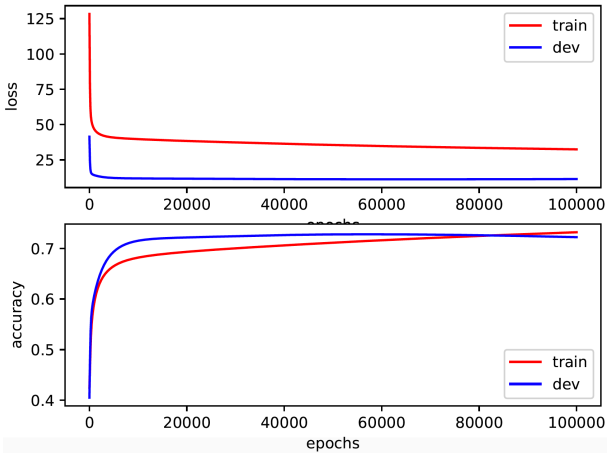

Fig 4. Loss and accuracy over train and dev sets for neural network

I chose the hidden layer to contain 15 neurons, a learning rate of 0.001, and 100,000 epochs of training. These parameters were chosen because increasing the number of neurons or increasing the learning rate beyond this point resulted in drastic overfitting of the training data.

Backward Feature Selection

For both logistic regression and GDA, the model using all formants as features had the highest accuracy. However, when testing each feature individually, I noted that F3 had significantly higher accuracy (0.6912 GDA and 0.7127 Logistic Regression) than either F1 (0.5187 GDA and 0.5187 Logistic Regression) or F2 (0.5325 GDA and 0.5399 Logistic Regression), suggesting that this formant may impart more predictive capacity upon the task.

Performance Metrics for Best Models

Below, I include confusion matrices as well as precision, recall, and F1 values for the best performing models, namely SVM with sigmoid and linear kernels, GDA, and logistic regression.

	1	0
1	0.2374	0.0863
0	0.1799	0.4964

	1	0
1	0.2518	0.1079
0	0.1655	0.4748

	1	0
1	0.2222	0.0370
0	0.1481	0.5926

	1	0
1	0.1851	0.0370
0	0.1851	0.5926

Table 2. Confusion matrices for (top left) GDA, (top right) logistic regression, (bottom left) SVM sigmoid, and (bottom right) SVM linear [predictions on left, true labels on top]

Model	Precision	Recall	F1
GDA	0.7333	0.5690	0.6408
Log_Reg	0.70	0.6034	0.6482
SVM_sig	0.8571	0.60	0.7059
SVM_lin	0.8333	0.50	0.6250

Table 3. Precision, recall, and F1 scores for best performing models

Discussion and Future Work

Overall, we can see that certain models perform fairly well in this task, achieving over 80% accuracy in the case of SVM with a sigmoid kernel. I do not report an oracle for this task

because this is not necessarily a conscious computation for humans. Thus, modeling a human oracle would include complex field linguistic experiments with a wide array of subjects.

Though I had originally hypothesized that deep learning would prove the optimal solution to this classification problem, I believe that the low dimension of the feature set and the size of my dataset contributed to an underperformance in the algorithm. In future work, I hope to explore using all formant values sampled over the duration of the vowel as feature vectors as opposed to only the average formant values as I have used here. Additionally, I plan on increasing the size of the dataset to attempt a more robust deep learning model while confirming my current results with regards to all algorithms.

With regards to the SVM results, given the recall of the SVM with a linear kernel, I do not believe that this is an optimal model for this problem. However, SVM with a sigmoid kernel performs better than all other models in almost all performance metrics, so this is the best classifier for the task at hand.

In the context of autosegmental theory, my results confirm that individual phonemes contain phonetic information about their environment, and that this information can be used as predictive capacity over the following consonants' place of articulation. Clearly, this model of phonetic language comprehension provides an explanation of how humans may lighten the cognitive load of such a daunting task. Additionally, interesting parallels could be drawn between methods of lightening cognitive load in the semantic, syntactic, and phonetic realms of linguistics. Ultimately, I believe that vowels carry even more information about their environment, and I hope to carry out multi-class prediction with a larger dataset to be able to classify both place *and* manner of articulation to further bolster autosegmental theory.

References

[1] Carrington, A. M., Fieguth, P. W., & Chen, H. H. (2014). A new Mercer sigmoid kernel for clinical data classification. *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. doi: 10.1109/embc.2014.6945092

[2] Freunberger, D., & Roehm, D. (2016). Semantic prediction in language comprehension: evidence from brain potentials. *Language, Cognition and Neuroscience*, *31*(9), 1193–1205. doi: 10.1080/23273798.2016.1205202

[3] Gow, D. W. (2003). Feature parsing: Feature cue mapping in spoken word recognition. *Perception & Psychophysics*, *65*(4), 575–590. doi: 10.3758/bf03194584

[4] Goldsmith, John. 1976a. *Autosegmental phonology*. Cambridge, MA: Massachusetts Institute of Technology dissertation.

[5] Hyman, L. M. (2014). How autosegmental is phonology? *The Linguistic Review*, *31*(2). doi: 10.1515/tlr-2014-0004

[6] Staub, A., & Clifton, C. (2006). Syntactic prediction in language comprehension: Evidence from either...or. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *32*(2), 425–436. doi: 10.1037/0278-7393.32.2.425

[7] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[8] CS229 Lecture Notes on the course website

My code for this project is available at the following link:
https://drive.google.com/open?id=10zRNH4TO6iLP33qCyp1A85pWqu3-Ij-Z