Comparative reconstruction of Middle Chinese with deep learning

Andrew Yang
Department of Linguistics
ycm@stanford.edu

Ken Hong
Symbolic Systems Program
kenhong@stanford.edu

Nattawat Luxsuwong
Department of Computer Science
nashlux@stanford.edu

December 13, 2019

Abstract

Using the reconstruction of Middle Chinese as our objective, we explore how linear and nonlinear relationships between syllabic features of modern-day Sinitic and Sino-Xenic vocabulary can be effectively approximated by various deep learning models. Furthermore, we show that leveraging character composition information can significantly help a neural network learn historical onsets and nuclei, supporting the theory that the construction of phonetic series can explain certain phonological patterns in Chinese.

1 Introduction

Our project aims to apply machine learning to comparative linguistic reconstruction. Many East Asian languages today have standardized pronunciations for Chinese characters. The character for 'middle' is pronounced /ʈʂɔŋ/ in Mandarin, /t͡sʊŋ/ in Cantonese, /tɕuŋ/ in Korean, etc. Using the known pronunciation of Chinese characters in modern Sinitic and Sino-Xenic languages, historical linguists have been able to create reconstructions of ancestors of modern Chinese languages thought to exist in the past two millenia.

However thorough a reconstruction scheme may be, it is inevitable that a historical linguist omits certain obscure characters from their system—after all, there are tens of thousands of Chinese characters, many of which were created after the ancestral language in question became extinct. By applying machine learning to linguistic reconstruction, we hope to teach a model the relationships between modern and archaic pronunciations so that we can extend the work that human linguists have done to Chinese characters outside the common scope of historical linguistic research. Further, we wish to test the capability for a machine learning model (which has no inherent rules) to learn a system like a reconstruction scheme (which, up until this point, has always been construed as rule-based).

For our project, we focus on reconstructions of Middle Chinese, an archaic prestige variety of Chinese spoken over a thousand years ago. We parsed Wiktionary to gather the pronunciations of over

15,000 Chinese characters in East Asian languages, specifically Mandarin, Cantonese, Japanese, and Korean, and aligned them to reconstructed pronunciations devised by influential historical linguists such as Bernhard Karlgren [4] and others. Because individual reconstruction schemes are not directly comparable, we focus on constructing our system to be aligned with Karlgren's system.

2 Related Work

The reconstruction of words that appear in proto-languages from which modern languages evolved can be considered one of the oldest problems in linguistics, with the *comparative method* being the standard technique for reconstruction. However, this is an extremely labor-intensive process that involves constructing extensive cognate sets and manually formulating phonological rules.

While past literature in applying computational methods to linguistic reconstruction is relatively sparse, two selected works that seek to faciliate the reconstruction of protolanguages are Bouchard-Côté 2013 [1] and Dekker 2018 [3]. Bouchard-Côté, whose work primarily focused on the Austronesian languages, proposed the use of probabilistic models of sound change coupled with the a Monte Carlo inference algorithm given a collection of cognate sets from modern languages. Dekker, however, proposes a different approach by training two neural network models (a recurrent neural network encoder-decoder and a structured perceptron) on pairs of words from two languages in order to detect sound correspondences.

Our project is one of the first to combine deep learning to modern pronunciation and character composition data to try to learn an established system of phonological rules.

3 Pre-Processing

Because the focus of our project is to use phonological data to predict reconstructed pronunciations, we crawled thousands of characters on Wiktionary for their pronunciations in Mandarin, Cantonese, Korean, and Japanese, as well as their reconstructed forms according to the Karlgren reconstruction.

3.1 Pronunciation data

After parsing Wiktionary for pronunciation, we ended up with examples like this:

Entry	Mandarin	Japanese	…	Korean	Reconst-*tone*	Reconst.
中	zhōng	ちゅう		jung	Level	ʈi̯uŋ

We began by separating these unstructured pronunciation data into a format separated into syllabic units, as is standard in the field of phonology. Each Chinese character is one syllable, so we separated pronunciations into *onset* + *nucleus* + *coda* (initial consonant, vowel, final consonant) for each language we are working with. For example, we would divide the unstructured pronunciation /ʈʂɔŋ/ into /ʈʂ/, /ɔ/, and /ŋ/. (For Sinitic languages, we also include the *tone* of a character.) Aside from phonological tradition, this was motivated by the fact that our dataset is quite small; we believe there is simply not enough data for an end-to-end model. Instead, our current architectures predict *onset*, *nucleus*, *coda*, and *tone* independently. Because these data are still categorical, we performed

one-hot encoding of each entry to get a sparse binary matrix with dimensions 15250×471. There were 36 onsets, 64 nuclei, 7 codas, and 4 tones. Below is a fragment of a processed entry.

Entry	Mand-*onset*	Mand-*nucleus*	…	Kor-*nucleus*	…	Reconst-*coda*
中	⟨zh⟩	⟨o⟩		⟨u⟩		/ŋ/

3.2 Phonetic components

Phonetic series are sets of characters hypothesized to have similar pronunciation. These characters are often phono-semantic compounds that often share the same phonetic component. Across modern Sinitic and Sino-Xenic, characters in these proposed phonetic series have been shown to have similar pronunciations [5], so we decided to collect phonetic series data as well. For example, the characters 吾, 语, and 梧 have similar pronunciations and share the phonetic component 五; as a result, they can be considered to be part of the same phonetic series.

We considered directly providing the hypothesized phonetic series for each character as too much of an advantage (it is almost an oracle); thus, we used character composition data available on Wiktionary as a proxy for the inferred phonetic series. By organizing characters in a directed acyclic graph, where arcs go from characters to their constituent characters, we filter out commonly found radicals (which have not been shown to influence pronunciation) and simply find for each character descendant with the most descendants of its own, as seen in Figure 1.

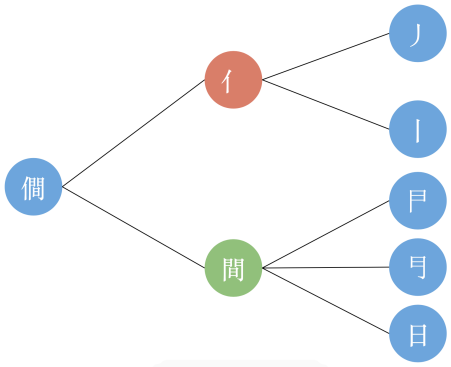

Figure 1: A sample section of the DAG

In this figure, we see that the non-radical descendant with the most children is 間 (in green); as a result, we assume that the phonetic component of 個 is 間 (which it is). This is done for every character in our dataset. We test the assumption that characters with similar mappings have similar pronunciations using our final model in Section 5.4.

4 Zero-Rule Baseline (0R)

We believed that a baseline model for our task should ignore relevant phonological attributes of the data. However, since our end product would hopefully be entire syllables, it makes sense that our

baseline model tries to reconstruct syllables as well. This way, we can compare the syllables generated by the baseline to the syllables generated by our improved models.

0R predicts the most frequent class every time. For each category (*onset*, *nucleus*, *coda*, *tone*), this baseline simply predicts the most frequent class. Below were our results:

onset	*nucleus*	*coda*	*tone*
8.37%	8.18%	40.14%	45.19 %

The relatively good performance on *coda* and *tone* is explained by the fact that our data had a large plurality of a specific label in both categories (`level` tone and `null` coda).

5 Architectures

Similar to the Zero-Rule baseline, each architecture we explored makes separate predictions for each category (*onset*, *nucleus*, *coda*, *tone*), with the exception of the alternate, quasi-baseline model we developed in section 5.1. The first three models use only pronunciation-based features, while the fourth one incorporates phonetic component information.

We divided our dataset of 15250 examples into 70% training and 30% testing. Further, we used 5-fold cross validation on the training set when building our models and only tested on the test set when it came time to evaluate. This helped prevent our models from overfitting and made them generalize to unseen data more effectively.

5.1 Horde of logistic regressions (LR-1)

To construct an alternative to 0R, we experimented with instantiating a logistic regression classifier for every one-hot encoded label category, such as `Karlgren_nucleus_i` or `tone_label_level`. In total, this amounted to 109 labels.

Each classifier is trained on the same training set to predict exactly one category, reducing the multiclass, multilabel problem to 109 binary classification tasks. For instance, the classifier assigned to `Karlgren_nucleus_i` would ideally learn to predict 1 or 0 for whether a training example has should have /i/ as its reconstructed *nucleus*.

5.2 One-versus-rest logistic regression (LR-2)

In this model, we divide the labels into *onset*, *nucleus*, *coda*, and *tone*. We also reduce the multiclass problem into a separate of binary classification problems, similar to LR-1. Each logistic regression classifier is compared against every other classifier for a particular category, with the moskt "confident" predicted label for each category being returned for each class. This ensures that exactly one *onset*, exactly one *nucleus*, etc. is predicted for each feature tuple.

5.3 Multilayer perceptrons (MLP-1)

The MLP-1 architecture consists of four separate neural networks that correspond to each label category. Each neural network has two 32-unit hidden layers with ReLU activation and an output layer with softmax activation. The number of nodes in the output layer matches the number of labels in the label category, so the MLP assigned to *onset* would have 36 output neurons while the one assigned to coda would have only 7. We determined hyperparameters were determined from a grid search over a wide array of possible values ranging from 1 to 50 epochs, $\eta \in [10^{-2}, 10^{-7}]$, and batch sizes of 1 to 4096. We found that, across the four networks, networks trained for 15 epochs with learning rate $\eta = 10^{-3}$ and batch size 256 converged faster.

5.4 Multilayer perceptrons with phonetic series (MLP-2)

Like MLP-1, the MLP-2 architecture consists of four separate networks each tasked with predicting a different label category. For these models, we introduced the processed phonetic component information (as a proxy for phonetic series) for each character and performed separate grid searches with cross-validation for each network, leading to four different sets of parameters. Out of over four thousand unique mappings to the hypothesized phonetic components, we discarded daughter nodes with fewer than three parent nodes, leaving around nine hundred additional sparse binary features.

5.4.1 MLP-2-*onset*

We finalized the architecture of MLP-2-*onset* as the following: input layer, 768-neuron hidden layer, 192-neuron hidden layer, output layer. The hidden layers both used ReLU activation and the output layer had softmax activation. During training, a drop-out rate of 0.3 was enforced between the two hidden layers, which was successful at reducing overfitting of the data. The model was trained for four epochs with learning rate $\eta = 10^{-3}$ and batch size of 256.

5.4.2 MLP-2-*nucleus*

We finalized the architecture of MLP-2-*nucleus* as the following: input layer, 1024-neuron hidden layer, 192-neuron hidden layer, output layer. The hidden layers both used ReLU activation and the output layer had softmax activation. During training, a dropout rate of 0.3 was enforced between the two hidden layers, which was successful at reducing overfitting of the data. The model was trained for four epochs with learning rate $\eta = 10^{-3}$ and batch size of 256.

5.4.3 MLP-2-*coda*

We finalized the architecture of MLP-2-*coda* as the following: input layer, 96-neuron hidden layer, 32-neuron hidden layer, output layer. The hidden layers both used ReLU activation and the output layer had softmax activation. The model was trained for five epochs with learning rate $\eta = 10^{-3}$ and batch size of 256.

5.4.4 MLP-2-*tone*

We finalized the architecture of MLP-2-*tone* as the following: input layer, 32-neuron hidden layer, 16-neuron hidden layer, output layer. The hidden layers both used ReLU activation and the output layer had softmax activation. The model was trained for ten epochs with learning rate $\eta = 10^{-3}$ and batch size of 256.

5.4.5 Insights

We note that the complexity of the chosen models for *onset* and *nucleus* is significantly greater than those for *coda* and *tone*. This is due to both the relatively large number of input and output neurons for these categories, as well as the increased difficulty in reconstructing the relevant phonological changes. When smaller models were applied to these categories, decreased performance was observed. We did not encounter this problem with *coda* and *tone*, which already had significantly better performance than the other two categories in LR-2 and MLP-1.

6 Results and Error Analysis

Our models performed quite well at predicting the *tone* and *coda* categories, with weaker performance in the *onset* and *nucleus* categories. The evaluation metric we used was raw accuracy; because historical reconstructions are based on precise phonological rules, we hold the belief that it is not enough to predict /p/ or /b/ when the true label is /p^h/; it is important that our models learn exactly the label and not something that is only phonologically similar. Our results are shown in Table 6 and Fig. 2.

	0R	LR-2	MLP-1	MLP-2
onset	8.37	59.80	70.86	75.71
nucleus	8.18	51.54	65.29	73.55
coda	40.14	90.49	92.28	92.59
tone	45.19	75.54	79.80	80.92

Table 1: Test accuracy percentages across four label categories for our models.

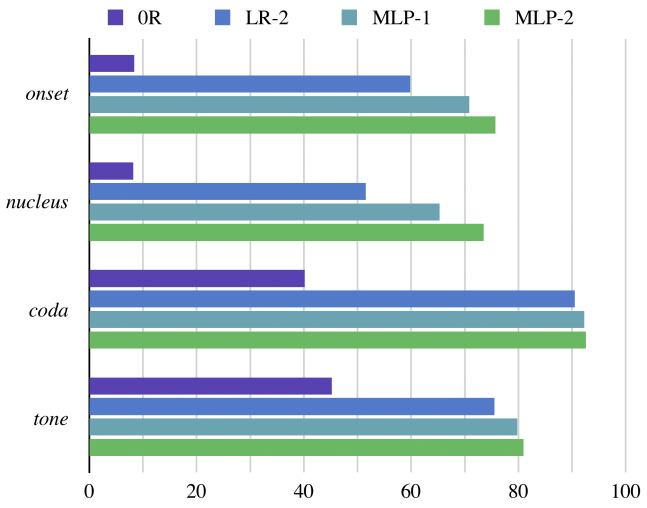

Figure 2: Test accuracies across our models.

6.1 LR-1

Because LR-1 (the horde of logistic regressions) was not guaranteed to predict exactly one *onset*, *nucleus*, *coda*, and *tone* for each example, we evaluated its performance by using raw accuracy in addition to *recall*. Since we do not segregate by label category, we simply analyze these statistics over all all labels in general.

Raw accuracy	.988
Mean recall	.470
Median recall	.516

Because labels were one-hot encoded, they are very sparse; each logistic regression classifier may encounter only a handful of 1s as training labels. As a result, many classifiers within LR-1 have a strong tendency to predict 0s all the time. This explains why the raw accuracy can be so much higher than the recall.

6.2 LR-2

The one-versus-rest approach with logistic regression performed very well on the *coda* and *tone* categories, but performed significantly worse on the *onset* and *nucleus* categories.

From a linguistics standpoint, this makes sense—the correspondence between codas in languages like Cantonese and Korean [2] strongly suggests that Middle Chinese codas were quite faithfully preserved. Furthermore, the correspondence between the third and second tones of Mandarin and Cantonese respectively, as well as the variety of tones in Cantonese also suggest that Middle Chinese codas can be reconstructed quite easily from existing data.

The lower performance on the *onset* and *nucleus* categories is also in line with our knowledge that onsets and nuclei display more instances of splits and mergers among East Asian languages. Furthermore, there are simply more onsets and nuclei to predict, whereas *coda* and *tone* are much smaller categories.

We note that achieving over 50% accuracy for *onset* and *nucleus* is still quite an impressive feat, given that there were 36 onsets and 64 nuclei to choose from.

6.3 MLP-1

Compared to LR-2, the MLP-1 architecture makes small improvements over *coda* and *tone* and makes more significant improvements in *onset* and *nucleus*. The small improvements were about what we had expected, since the relationships between coda and tone in Middle Chinese and modern languages is more linear than that of onset and nucleus. As a result, changing logistic regression to a neural network will not lead to drastic improvements in performance on *coda* and *tone*.

The fact that the prediction accuracy for *onset* and *nucleus* each improved by over 10% supports our intuition that a nonlinear model may be more apt for learning the relatively complex phonological correspondences for onsets and nuclei.

6.4 MLP-2

By introducing phonetic component data to each of the four networks from MLP-1 and conducting a thorough hyperparameter search, we were able to achieve notable gains in performance.

Again, we observed very slight performance increases in the *coda* and *tone* categories. This reinforces the notion that these two categories were already quite linear to begin with, so adjusting hyperparameters would not have a large effect on improving performance. Moreover, prior knowledge of the Mandarin and Cantonese languages tells us that the phonetic components of characters hardly serve to inform speakers about the correct tone, which is supported by the observation that performance for *tone* increased by only one percent.

The main performance boosts occurred in the *onset* and *nucleus* categories, which saw 5% and 8% increases in accuracy, respectively. This is in line with the hypothesis that knowing about the phonetic component of a character allows one to associate that characters with other characters with similar pronunciations.

6.4.1 Misclassfications

After testing MLP-2 on our test set, we further analyzed the predictions made by MLP-2-*onset* and MLP-2-*nucleus* because *onset* and *nucleus* are naturally more difficult to predict. Confusion matrices for these two categories can be found in Appendix A, Figures 3 and 4. Table 2 shows the ten labels most likely to be misclassified in the *onset* and *nucleus* categories.

onset	Predicted	d͡ʑʰ	pʰ	pʰ	ʂ	t͡ɕʰ	ʈʰ	z	bʱ	t͡s	t͡ʂ
	Actual	ʑ	p	bʱ	s	t͡ɕ	t͡ɕʰ	s	p	d͡zʱ	t͡ɕ
nucleus	Predicted	i̯ɑ	wă	wə	wə	iwei	wăi	ɑi	i̯ɛi	wɐ	wɐ
	Actual	i̯u	uɑ	wæ	uo	iei	uăi	ăi	iei	wɑ	wæ

Table 2: Frequently misclassified labels.

These results seem very reasonable—the sounds in each misclassified pair are phonetically similar. For instance, the pair (d͡ʑʰ, ʑ) differ only by affrication and aspiration. The pairs of misclassified nuclei, such as the phonetically similar (wă, uɑ) and (i̯ɛi, iei) also make sense.

7 Future Work

Given more time, we would like to more accurately parse the modern East Asian languages that we used as features. For instance, when we were separating Mandarin syllables into their constituent parts, we used the Romanized strings as a proxy for actual, contrastive phonetic elements. Although it works fine most of the time, this naive assumption causes us to analyze the ⟨o⟩ in ⟨bo⟩ and ⟨dong⟩ as the same nucleus when they are in fact highly contrastive. We would also like to refine the method for inferring phonetic components—instead of prioritizing component characters with many daughter component characters, we can consider using *stroke count* instead.

Moreover, we would like to experiment with multitask learning as an alternative to constructing four separate models for each architecture. We want to see if we can exploit similarities in, say, the prediction of *onset* and *nucleus*, that can enable a network to predict multiple categories at once.

8 Conclusion

We conclude that deep learning is a viable way to learn sound correspondences between cognates. Our findings demonstrate that characteristics like *coda* and *tone* can be learned relatively well by simpler models like logistic regression, whereas more complex characteristics like *onset* and *nuclei* are better approximated using a combination of pronunciation data, character composition data, and careful feature engineering. We hope that this work can be applied to a few different areas in the study of East Asian languages, such as phonological change and orthography.

9 Contributions

Andrew parsed the Karlgren reconstructions and character composition data for each character and developed the baseline, logistic regression, and first neural network model. Ken scraped pronunciation data from Wiktionary, featurized characters for modern languages, and developed the second neural network model which incorporates phonetic series data. Nattawat developed the cross-validation pipeline, computed error metrics for each model, and produced visualizations of the data. All three members contributed to the progress report, poster, and final report.

Our code can be accessed at `github.com/ycm/cs221-proj`.

References

[1] A. Bouchard-Côté, D. Hall, T. L. Griffiths, and D. Klein. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*, 110(11):4224–4229, 2013.

[2] P. Chu. Onset, rhyme and coda corresponding rules of the sino-korean characters between cantonese and korean. 2008.

[3] P. Dekker. Msc thesis: Reconstructing language ancestry by performing word prediction with neural networks. 01 2018.

[4] B. Karlgren. The reconstruction of ancient chinese. *T'oung Pao*, 21(1):1–42, 1922.

[5] B. Karlgren. *Grammata serica recensa*, volume 29. Museum of Far Eastern Antiquities, Stockholm, 1957.

A Appendix

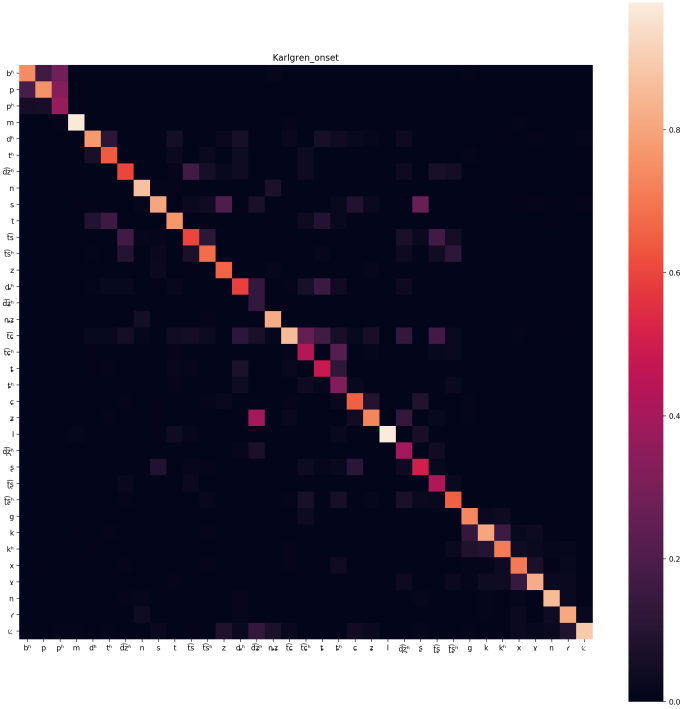

Figure 3: Confusion matrix for MLP-2-*onset*.

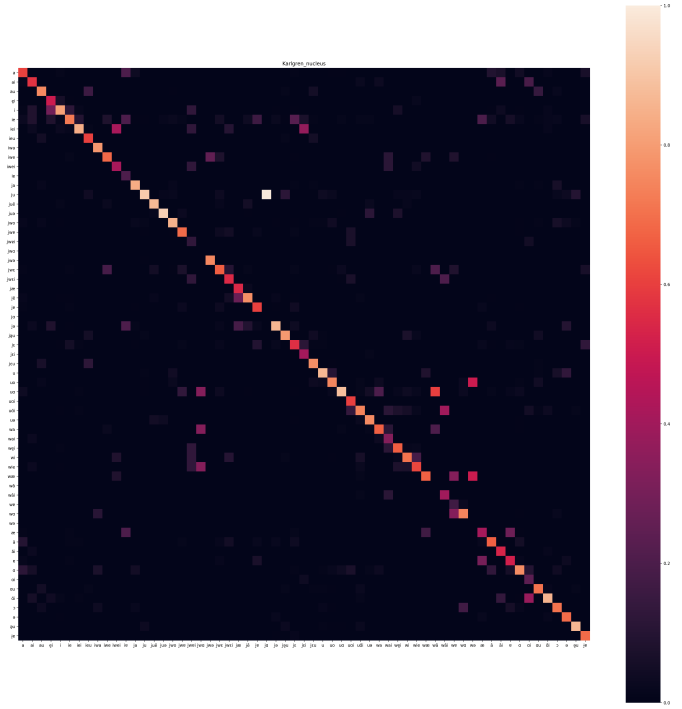

Figure 4: Confusion matrix for MLP-2-*nucleus*.