Annotating Farsi vowels using a Tajik corpus

Hanzhi Zhu
hanzhiz@stanford.edu

Abstract

A challenge facing any language with a non-phonemic writing system is how to convert written text into a fuller, phonetic representation. In this project, I explore the task of inserting vowels into Farsi text written in the Arabic script, which does not fully mark vowels. Given the lack of publicly available datasets or dictionaries which provide a mapping from Farsi Arabic-script text or words to a phonetic transcription, I use corpus in Tajik, a related language written in a fully vocalized alphabet. Despite the differences between the two languages, a classifier trained on the Tajik corpus produces noticeable results for Farsi vowel insertion.

1 Introduction

This project seeks to address two issues: the vowel insertion problem for Farsi, as well as the feasibility of using a model trained on one language (Tajik) to be applied to another (Farsi).

1.1 Farsi vowel insertion

In this paper, I using the term *Farsi* to refer to the formal standard variety of the Persian language as used in Iran, whereas Persian refers to the language in a larger historical and geographic context. Iranian Farsi is written using a modified version of the Arabic script. There are several properties of the Perso-Arabic writing system that deviate from a true phonemic/phonetic representation. First, Farsi preserves Arabic loanwords in its original form, resulting in several consonant letters in the writing system which map onto only one phonemic in Farsi, (e.g. four letters map to /z/). Second, the original Persian short vowels (/æ/, /e/, /o/ in modern Farsi) and the gemination of consonants are represented using diacritics which are almost never used in practice, expect sometimes for the sake of disambiguation. In theory, every character in Farsi text is assigned one of the three vowel diacritics or a null diacritic; word-initial vowels are represented by a the letter *alif*, which is silent word-initially. Third, several Farsi letters are ambiguous as to whether they represent vowels or consonants. The letter <y> can represent /i/, and /y/ and <w> can represent /u/, /o/, and /v/.

For this project, I define the vowel insertion task as annotating the short vowels in diacritic-less Farsi text. This definition suffices to disambiguate even the long vowels, as short vowel diacritics combine with the letters <y> and <w> to form long vowels. One Farsi long vowel, /a:/ is never ambiguous.

1.2 Comparison with Tajik

The Tajik language is a variety of Persian spoken in Tajikistan and written with the Cyrillic alphabet. As it is represented near-phonemically all vowels are fully written. Furthermore, a separate letter for /i/ is used word-finally to indicate that it is not the *ezafe* construction, providing morphological information. However, the most salient difference between Farsi and Tajik is in fact in their vowel systems. Eight original Persian vowels merged into six Farsi and Tajik vowels, but with different mergers, as show in 1.

Old Persian	a	a:	i	i:	e	u	u:	o
Farsi	æ	a	e	i	i	o	u	u
Tajik	æ	a	i	i	e	u	u	ö

Table 1: Persian, Farsi, and Tajik vowels

Despite these different correspondences, a Tajik

word token combined with an equivalent Farsi token is enough to generate almost all the Farsi vowels.

Given that Tajik is so closely related to Farsi, one would expect that a Tajik corpus would be almost functionally equivalent to a vowel-annotated Farsi corpus. Many methods would be available to take advantage of this similarity.

2 Prior work

In fact, several recent studies have attempted to make use of similarity of closely-related languages, most of which use finite-state transducers. Few work has been done on the Farsi-Tajik pair. Among the few, Megerdoomian and Parvaz (2008) use a finite-state transducer to convert Tajik text into Farsi in order to bootstrap a machine translation system for Tajik using existing Farsi MT models. On the other hand, **?** explored using a character alignment system using a hand-generated dataset to convert between Tajik and Farsi words. Much work has also been done on a similar language pair, Hindi and Urdu (Malik et al., 2009).

Previous work on vowel insertion in Farsi has been successful due to their use of dictionaries which map Farsi text to a phonetic representation. For example, Nojoumian (2011) uses a hand-built dictionary with Farsi text directly annotated for diacritics in order to solve the diacritic insertion task. Hendessi et al. (2005) work on a speech synthesizer for Farsi, which relies on a Farsi dictionary with phonetic transcriptions. Although their speech synthesizer is superficially a different problem, vowel insertion is a sub-task of their system. As none of these data were available to me, I was forced to approach the vowel insertion problem from a different angle. As both these existing systems perform at the word level, they are not suitable to handle out-of-vocabulary words. I built a system which predicts vowels at the subword level, and thus does not take whole word tokens into account. Although I cannot evaluate a hybrid model containing an existing Farsi vowel insertion system and my system, I leave it to future work that such a combination would offer improved performance on the Farsi vowel insertion task.

3 Data and Methods

I describe a system I implemented in which, given an input Farsi text, I generate a file in which vowel diacritics are added. I approach this by considering for each letter in the Farsi text whether there should be a vowel diacritic added to that letter, and if so, which one.

3.1 Corpora

In order to create training and testing examples, I needed a sizeable Tajik corpus and a small Farsi corpus.

I downloaded a publicly available Tajik corpus of separate sentences news text and user comments (2 million words) and converted it into an equivalent ASCII representation. I discarded all sentences which did not include a Tajik-specific letter, as these sentences would usually overload a Cyrillic letter to stand for another, due to keyboard limitations. For example, /h/ and /x/ have separate Tajik Cyrillic letters, but are written both as <x> for users without a Tajik keyboard layout. Although this corpus would not be considered large for word-level tasks, I needed to take only the first tenth of the corpus in order to be able to train on my character-level task.

In order to generate a small Farsi corpus, I downloaded three documents from the UniPers website, a website about a proposed Roman orthography for Tajik. These documents were available in both Farsi Arabic orthography as well as a Latinate transcription. I collected the Arabic-script documents and converted them into an equivalent ASCII representation. Since I needed to add the vowel diacritics to the documents in order to evaluate my results, I wrote a Python script to heuristically insert vowel diacritics by attempting to align the transcription with the Arabic-script words.

3.2 Training and testing examples

In order to generate the positive training examples, I created a feature vector for each vowel in the Tajik corpus. As I intended for this model to be applied to the Farsi data, I hand-crafted the features in such a way that they would be as consistent as possible, and used regular expression rewrite rules to make the Tajik and Farsi word tokens share the same set of consonantal letters

that would be used as features. For each Tajik vowel, I took the previous and following two consonantal letters as features, as well as the count of how many consonant letter appear before and after the vowel. Instead of directly predicting the Farsi vowel diacritic, I let the label classes be the Tajik vowel letters instead, and applied regex rewrite rules to convert the Tajik vowel letters to the three Farsi diacritics.

I generated negative training examples by considering the middle of every two adjacent consonants as well as word-final consonants. I encoded the features exactly as if there was a vowel between the two consonants.

To run on the Farsi corpus, I considered each letter at a time and generated a feature vector to predict whether the following letter in the Tajik model would be a vowel.

My Tajik vowel model was created by training a Multinomial Naïve Bayes classifier using the scikit-learn package on the Tajik training data. I run this model on the Farsi corpus by considering each Farsi letter at a time, generating a feature vector to predict whether the following letter in the Tajik model would be a vowel. To evaluate, I applied regular expression rewrite rules to convert the Tajik vowels in to Farsi form base on context, and ran a script to check each vowel and each absence of a vowel after a Farsi consonantal letter. I generated precision and recall scores by considering all instances in which a vowel was predicted to be positive, and in which a vowel was not predicted to be negative.

4 Results

4.1 Baseline

I considered a simple baseline in which one guesses that no vowels should be inserted anywhere in the Farsi text. Such a baseline was able to achieve 59.3% accuracy, but with 0 precision and recall.

With my classifier, I was able to achieve a 10% boost on accuracy, bringing it up to 70.1%. Precision and recall percentages of more than 50% were achieved, shown below.

Although these results are nowhere near perfect and would not match against the results of systems which use Farsi pronouncing dictionaries, they demonstrate an interesting result: not only can we predict Farsi vowel diacritics just from the information encoded in Tajik text, we can do so without regard to matching two equivalent words, but instead by simply considering the character-level context. This result can easily be applied to improve existing Farsi vowel insertion systems by allowing a classifier to run on unseen vocabulary items.

	P	R	F	Acc
Baseline	0	0	0	59.3
MNB	66.3	53.6	59.3	70.1

Table 2: resultvalues

After examining the results, I found that the most common errors occurred around a Farsi <w> or <y>, as these are letters which could be either a consonant or a vowel in Tajik.

4.2 Future work

To improve the task that I explore in this paper, I would like to incorporate more features than the very basic set that I implemented. A bag-of-characters feature might boost performance, as well as features that consider word-contextual information by including features from the previous or following words. Ultimately, I would hope to gain access to a Farsi pronouncing dictionary based model and experiment with building a hybrid model combining it with mine.

5 Conclusion

Two conclusions can be drawn from the positive results of my project. First, the vowel insertion task can be improved on by statistical methods which would be able to handle out-of-vocabulary words in a commercial vowel insertion or speech synthesis system. Second, my results show that despite the large orthographic differences between the two language varieties of Farsi and Tajik, the closeness of their underlying linguistic form allow for one language to aid the other on natural language processing tasks.

References

Hendessi, F., Ghayoori, A., and Gulliver, T. A. (2005). A speech synthesizer for persian text using a neural network with a smooth ergodic

hmm. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(1):38–52.

Malik, A., Besacier, L., Boitet, C., and Bhattacharyya, P. (2009). A hybrid model for urdu hindi transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 177–185. Association for Computational Linguistics.

Megerdoomian, K. and Parvaz, D. (2008). Low-density language bootstrapping: the case of tajiki persian. In *LREC*.

Nojoumian, P. (2011). *Towards the Development of an Automatic Diacritizer for the Persian Orthography based on the Xerox Finite State Transducer*. PhD thesis, University of Ottawa.