Procesamiento del Lenguaje Natural, Revista nº 52 marzo de 2014, pp 13-20 recibido 14-11-13 revisado 30-01-14 aceptado 03-02-14

Learning to map variation-standard forms in Basque using a limited parallel corpus and the standard morphology

Aprendizaje de correspondencias variante-estándar usando un corpus paralelo limitado y la morfología del estándar

Izaskun Etxeberria*, Iñaki Alegria*, Mans Hulden, Larraitz Uria***
*IXA group, UPV-EHU. M. Lardizabal 1, 20018 Donostia
**The Department of Modern Languages. PO Box 24, 00014 University of Helsinki
izaskun.etxeberria@ehu.es, i.alegria@ehu.es, mans.hulden@helsinki.fi, larraitz.uria@ehu.es

Resumen: Este artículo explora tres diferentes métodos de aprendizaje de las variantes de un idioma (formas dialectales o diacrónicas) a partir de un pequeño corpus paralelo suponiendo que la morfología estándar está disponible.
Palabras clave: normalización léxica, morfología, bibliotecas digitales

Abstract: This paper explores three different methods of learning to map variant word form (dialectal or diachronic) to standard ones from a limited parallel corpus of standard and variant texts, given that a computational description of the standard morphology is available.
Keywords: lexical normalization, morphology, digital libraries

1 Introduction

In our work with the Basque language, a morphological description and analyzer is already available for the standard language, along with other tools for processing the language (Alegria et al., 2002). However, it would be convenient to be able to analyze variant forms as well. As the dialectal differences within the Basque language are largely lexical and morphophonological, analyzing the dialectal forms would require a separate morphological analyzer able to handle the unique lexical items in the dialect together with the differing affixes and phonological changes. Likewise, diachronic variants can not be analyzed by standard morphological analyzers and stemmers. For example, when searching in digital libraries containing old texts, it is impossible to find the corresponding old forms for a modern word without linguistic knowledge.

Morphological analyzers are traditionally hand-written by linguists, most commonly using some variant of the popular finite-state morphology approach (Beesley and Karttunen, 2002). The construction of an analyzer entails having an expert who models a lexicon, inflectional and derivational paradigms as well as phonological alternations, and then producing a morphological analyzer/generator in the form of a finite-state transducer.

As the development of such wide-coverage morphological analyzers is labor-intensive, the hope is that an analyzer for a variant could be automatically learned from a limited parallel standard/variant corpus, given that an analyzer already exists for the standard language. This is an interesting problem because a good solution to it could be applied to many other tasks as well: to enhance access to digital libraries (containing diachronic and dialectal variants), for example, or to improve the processing of informal registers such as microblogging texts (some techniques described here have been used in our participation on the TweetNorm_es shared task at SEPLN2013).[1]

In this paper we evaluate three methods to learn a model from a standard/variant parallel corpus that translates a given word of the dialect to its equivalent standard-form (called *Batua*). All the methods are based on finite-state phonology. The first two methods have been previously reported (Hulden et al., 2011).

In this context, the use of statistical machine translation (SMT) technology is not adequate since there is not any big parallel corpus available.

The variant we have so far used for our ex-

[1] http://www.congresocedi.es/images/site/actas/ActasSEPLN.pdf

ISSN 1135-5948

periments is Lapurdian,[2] a dialect of Basque spoken in the Lapurdi (fr. Labourd) region in the Basque Country. As Basque is an agglutinative and highly inflected language, we believe some of the results can be extrapolated to many other languages facing similar challenges.

The differences between the dialect and the standard are minor overall; the word order and syntax are usually unaffected, and only a few lexical items differ. However, even such relatively small discrepancies cause great problems in the potential reuse of current tools designed for the standard forms.

We have experimented with three approaches that attempt to improve on a simple baseline of memorizing word-pairs in the dialect and the standard.

The first approach is based on the work by Almeida et al. (2010) on contrasting orthography in Brazilian Portuguese and European Portuguese. In this approach, differences between substrings in distinct word-pairs are memorized and these transformation patterns are then applied whenever novel words are encountered in the evaluation. To prevent overgeneration, the output of this learning process is later subject to a morphological filter where only actual standard-form outputs are retained.

The second approach is an Inductive Logic Programming-style (ILP) (Muggleton and De Raedt, 1994) learning algorithm where phonological transformation rules are learned from word-pairs. The goal is to find a minimal set of transformation rules that is both necessary and sufficient to be compatible with the learning data, i.e. the word pairs found in the training data.

The third approach uses *Phonetisaurus*, a weighted finite state transducer (WFST) driven phonology tool (Novak et al., 2012) in order to learn the changes using a noisy channel model. Based on the improved results using *Phonetisaurus*, we decided to explore morphophonological changes rather than phonological ones.

The paper is organized as follows. Section 2 describes the related work. The characteristics of the corpus used for our experiments are described in section 3. Sections 4 and 5 describe the steps and variations of the methods we have applied and how they are evaluated. Section 6 presents the experimental results, and finally, section 7 discusses the results and presents possibilities for potential future work in this field.

[2]Sometimes also called Navarro-Labourdin or Labourdin.

2 Related work

The general problem of supervised learning of dialectal variants or morphological paradigms has been discussed in the literature with various connections to computational phonology, morphology, machine learning, and corpus-based work. For example, Kestemont et al. (2010) presents a language-independent system that can 'learn' intra-lemma spelling variation.

Koskenniemi (1991) provides a sketch of a discovery procedure for phonological two-level rules. The idea is to start from a limited number of paradigms, essentially pairs of input-output forms, where the input is the surface form of a word and the output a lemmatization plus analysis.

Mann and Yarowsky (2001) present a method for inducing translation lexicons based on transduction models of cognate pairs via bridge languages. Bilingual lexicons within language families are induced using probabilistic string edit distance models. Inspired by that paper, Scherrer (2007) uses a generate-and-filter approach quite similar to our first method. He compares different measures of graphemic similarity applied to the task of bilingual lexicon induction between Swiss German and Standard German.

3 The corpus and the baseline

3.1 The corpus

The parallel corpus used in this research was built in the *TSABL* project developed by the IKER research group in Baiona (fr. Bayonne).[3] It contains sentences written in the Lapurdian dialect as well as their equivalent sentences in standard Basque.

Table 1 presents the details of the corpus, which consists of 2,117 parallel sentences, totaling 12,150 words (roughly 3,600 types). In order to provide data for our learning algorithms as well as to test their performance, we have divided the corpus into two parts: 80% of the corpus is used for the learning task (1,694 sentences) and the remaining 20% (423

[3] *Towards a Syntactic Atlas of the Basque Language*, web site: `http://www.iker.cnrs.fr/--tsabl-towards-a-syntactic-atlas-of-.html`

	Corpus	Dev	Test
Sentences	2,117	1,694	423
Words	12,150	9,734	2,417
Unique words			
Standard Basque	3,553	3,080	1,192
Lapurdian	3,830	3,292	1,239
Filtered pairs	3,610	3,108	1,172
Identical pairs	2,532	2,200	871
Distinct pairs	1,078	908	301

Table 1: Characteristics of the parallel corpus used for experiments.

sentences) for the evaluation. As the data show, roughly 23% of the word-pairs are distinct.

3.2 The baseline

The baseline of our experiments is a simple method, based on a dictionary which contains a list of correspondences among words extracted from the learning portion (80%) of the corpus. This list of correspondences contains all different word pairs in the variant vs. standard corpus. The baseline approach consists simply of memorizing all the distinct word pairs seen between the dialectal and standard forms, and subsequently applying this knowledge during the evaluation task. That is, if an input word during the evaluation has been seen in the training data, we provide the corresponding previously known output word as the answer.

4 Previous work

In our previous work, we employed two different methods to produce an application that attempts to extract generalizations from the training corpus to ultimately be able to produce the equivalent standard word corresponding to a given variant word.

The first method is based on already existing work by Almeida et al. (2010) that extracts all substrings from lexical pairs that are different. From this knowledge we then produce a number of phonological replacement rules that model the differences between the input and output words. In the second method, we likewise produce a set of phonological replacement rules, using an ILP approach that directly induces the rules from the pairs of words in the training corpus. The core difference between the two methods is that while both extract replacement patterns from the word-pairs, the first method does not consider negative evidence in formulating the replacement rules. Instead, the existing morphological analyzer is used as a filter after applying the rules to unknown text to prevent overapplication. The second method, however, uses negative evidence from the word-pairs in delineating the replacement rules as is standard in ILP-approaches, and the subsequent morphological filter for the output plays much less of a role.

4.1 Format of rules

Two of the evaluated methods involve learning a set of string-transformation rules to convert words, morphemes, or individual letters (graphemes) in the dialectal forms to the standard variant. The rules that are learnt are in the format of so-called phonological replacement rules (Beesley and Karttunen, 2002) which we have later converted into equivalent finite-state transducers using the freely available *foma* toolkit (Hulden, 2009). The reason for this conversion of the rule set to finite-state transducers is twofold: first, the transducers are easy to apply rapidly to input data using available tools, and second, the transducers can further be modified and combined with the standard morphology already available to us as a finite transducer.

In its simplest form, a replacement rule is of the format

$$A \rightarrow B \,||\, C \_ D \tag{1}$$

where the arguments A,B,C,D are all single symbols or strings. Such a rule dictates the transformation of a string A to B, whenever the A is found between the strings C and D. Both C and D are optional arguments in such a rule, and there may be multiple, comma-separated, conditioning environments for the same rule.

For example, the rule:

```
h -> 0 || p _ , t _ , l _ , _ a s o
```
(2)

would dictate a deletion of h in a number of contexts; when the h is preceded by a p, t, or l, or succeeded by the sequence *aso*, for instance transforming **ongiethorri** (Lapurdian) to **ongietorri** (Batua).

4.2 Method 1 (lexdiff) details

The first method is based on the idea of identifying sequences inside word pairs

where the output differs from the input. This was done through the already available tool *lexdiff* which has been used in automatic migration of texts between different Portuguese orthographies (Almeida et al., 2010). The *lexdiff* program tries to identify sequences of changes from seen word pairs and outputs string correspondences such as, for example: `76 ait -> at ; 39 dautz -> diz` (stemming from pairs such as (**joaiten/joaten** and **dautzut/dizut**), indicating that `ait` has changed into `at` 76 times in the corpus, etc., thus directly providing suggestions as to phonologically regular changes between two texts, with frequency information included.

With such information about word pairs we generate a variety of replacement rules which are then compiled into finite transducers with the *foma* application. Even though the *lexdiff* program provides a direct string-to-string change in a format that is directly compilable into a phonological rule transducer, we have experimented with some possible variations of the specific type of phonological rule we want to output:

- We can restrict the rules by frequency and require that a certain type of change be seen at least n times in order to apply that rule.
- We can limit the number of rules that can be applied to the same word.
- We can control the application mode of the rules: sequential or parallel.
- We can compact the rules output by *lexdiff* by eliminating redundancies and constructing context-sensitive rules. For example: given a rule such as `rkun -> rpen`, we can convert this into

  `k u -> p e || r _ n` (3)

  This has a bearing on the previous point and will allow more rewritings within a single word in parallel replacement mode since there are fewer characters overlapping.

Once a set of rules is compiled with some instantiation of the various parameters discussed above and converted to a transducer, we modify the transducer in various ways to improve on the output.

Firstly, we restrict the output from the conversion transducer to only allow those words as output that are legitimate words in standard Basque.

Secondly, in the case that even after applying the *Batua* filter we retain multiple outputs, we simply choose the most frequent word.

4.3 Method 2 (ILP) details

The second method we have employed works directly from a collection of word-pairs (dialect/standard in this case). We have developed an algorithm that from a collection of such pairs seeks a minimal hypothesis in the form of a set of replacement rules that is consistent with all the changes found in the training data. This approach is generally in line with ILP-based machine learning methods (Muggleton and De Raedt, 1994). However, in contrast to the standard ILP, we do not learn statements of first-order logic that fit a collection of data, but rather, string-to-string replacement rules.

The two parameters to be induced are (1) the collection of string replacements $X \rightarrow Y$ needed to characterize the training data, and (2) the minimal conditioning environments for each rule, such that the collection of rules model the string transformations found in the training data.

The procedure employed for the learning task is as follows:

(1) Align all word pairs (using minimum edit distance by default).

(2) Extract a collection of phonological rewrite rules.

(3) For each rule, find counterexamples.

(4) For each rule, find the shortest conditioning environment such that the rule applies to all positive examples, and none of the negative examples. Restrict rule to be triggered only in this environment.

The following simple example should illustrate the method. Assuming we have a corpus of only two word pairs:

emaiten	ematen
igorri	igorri

From this data we would gather that the only active phonological rule is $i \rightarrow \emptyset$, since

all other symbols are unchanged in the data. However, we find two counterexamples to this rule (step 3), namely two *i*-symbols in *igorri* which do not alternate with ∅. The shortest conditioning environment that accurately models the data and produces no overgeneration (does not apply to any of the *i*s in *igorri*) is therefore:

```
i -> ∅ || a _                (4)
```

the length of the conditioning environment being 1 (1 symbol needs to be seen to the left plus zero symbols to the right).

5 *Learning WFSTs using the noisy channel model*

We wanted to test the use of WFSTs (very popular in speech technology) in the task of learning Lapurdian/Standard Basque in order to obtain new methods and results and to compare them with the previous ones.

The first tool we used was Carmel[4] but the results obtained were worse than the previous ones (and the tuning process was very challenging).

Then, we experimented with a more modern tool for the purpose. The Phonetisaurus[5] tool was presented at the FSMNLP workshop of 2012 by J. Novak as a WFST-driven grapheme-to-phoneme (g2p) framework suitable for rapid development of high quality g2p or p2g systems. It is a new alternative, open-source, easy-to-use and authors report promising results.

The framework include three functions: (1) Sequence alignment, (2) Model training and, (3) Decoding (Novak et al., 2012).

The alignment algorithm is capable of learning many-to-many relationships and include three modifications to the basic toolkits: (a) a constraint is imposed such that only many-to-one and one-to-many alignments are considered during training. (b) During initialization a joint alignment lattice is constructed for each input entry, and any unconnected arcs are deleted. (c) All arcs, including deletions and insertions are initialized to and constrained to maintain a nonzero weight.

The model training works as following: (a) Convert aligned sequence pairs to sequences of aligned joint label pairs, (b) Train an n-gram model from (a); (c) Convert the n-gram model to a WFST. Step (c) may be performed with any language modeling toolkit.

The default decoder provided by the distribution simply extracts the shortest path through the phoneme lattice created via composition with the input word.

5.1 Using Phonetisaurus

We have used the Phonetisaurus tool to obtain a grapheme-to-grapheme system, i.e. not a g2p or p2g tool. In practice, applying the tool is straightforward and can be described in two steps:

1. Prepare the data from which the model has to learn. In our case, this is a dictionary of word pairs that have been obtained from the corpus collecting equal and different word pairs such as: **izan / izan, guziek / guztiek**, and so on.
2. Train a model using this data. A Language Model training toolkit is necessary in this step for the *n*-gram calculations, and there are different possibilities as the author mentions in the tutorial (*mitlm, NGramLibrary, SRILM, SRILM MaxEnt extension, CMU-Cambridge SLM*). We used *NGramLibrary* for our experiments.

Once the model has been trained and converted to a WFST format, it can be used to generate correspondences for previously unseen words. In contrast to the *Carmel* tool, it is not necessary to infer an FST because the tool builds it through the alignments and *n*-gram training process. Only the data needs to be supplied in the appropriate format.

There are two parameters to fix when we ask to the WFST to generate correspondences for new words: the number of transductions the WFST is going to return for each word and the size of the search beam.

As is usual in *n*-gram based decoding, increasing the search beam evaluates more hypotheses but at a cost of decoding speed. The default value leads to a reduced number of hypotheses.

We carried out a tuning process to decide the best values for those parameters. In order to perform the experiments, we divided the development corpus (the 80% of the total corpus) into 4 complementary subsets to

[4] http://www.isi.edu/publications/licensed-sw/carmel

[5] http://code.google.com/p/phonetisaurus/

apply a cross-validation technique looking for the best values of the mentioned parameters. Dividing the corpus into four subsets allows us to make four experiments in which the test subset is the same size as the the final test. The conclusions of those experiments are presented in section 6.2.

When there are multiple answers for a corresponding variant, it becomes necessary to perform some filtering. The first filter is obvious: we eliminate the answers that do not correspond to accepted standard words. Between the rest of the words, we select the most probable answer, according to *Phonetisaurus*.

In total, we have performed three different experiments with *Phonetisaurus* giving different pairs to learn.

5.2 Word-word and word-morphemes pairs

In the next three subsections the three different experiments are presented.

5.2.1 word-word

In the first experiment, we have provided the tool with all the word pairs obtained from the development corpus including identical pairs and distinct pairs. For example:

emaiten → e m a t e n
nehori → n e h o r i

5.2.2 word-morphemes

In the second and the third experiments we provide *Phonetisaurus* with different pairs to train on. In the second part of the dictionary we have marked the morphological analysis of the corresponding standard word instead of the word itself. Using the morphological analysis we have performed two experiments.

- In the first one, the analysis includes morphophonemes and diacritics. For the words above, this would look like:

  emaiten → e m a N + t e n
  nehori → n e h o Q + R i

  Here, **N**, **Q** and **R** are morphophonemes expressing epenthetic **n**, **r** in lemmas, and **r** in suffixes.

- In the second experiment the analyses have been slightly simplified by converting morphophonemes to their equivalent grapheme form and by deleting diacritics. The result is the concatenation of the morphemes using their canonical forms. For the words above this would be:

  emaiten → e m a n + t e n
  nehori → n e h o r + r i

  The hypothesis is that some morphophonemes and diacritics have a very low probability and are difficult to integrate into the learning process.

In both experiments, due to the WFST generating a fixed number of candidates for a dialectal word, which in this case are morphological analyses, a new step is necessary in order to find the corresponding standard forms. In addition, an analysis may generate more than one standard form, and in this case we have to select only one of them (the most frequent one in our implementation).

6 Evaluation and results

We have measured the quality of the different approaches by the usual parameters of precision, recall and the harmonic combination of them, the F_1-score, and analyzed how the different options in the approaches affect the results.

For the WFST solution, three different runs have been evaluated corresponding to the three possible representations of the standard form: word, morpheme sequence, and simplified morpheme sequence.

As mentioned above, the learning process has made use of 80% of the corpus, leaving 20% of the corpus for evaluation of the above-mentioned approaches. In the evaluation, we have only tested those words in the dialect that *differ* from words in the standard (which are in the minority). In total, in the evaluation part, we have tested the 301 words that differ between the dialect and the standard in the evaluation part of the corpus.

The results for the baseline—i.e. simple memorization of word-word correspondences—are (in %): P = 95.62, R = 43.52 and F_1 = 59.82. As expected, the precision of the baseline is high: when the method gives an answer it is usually the correct one. But the recall of the baseline is low, as is expected: slightly less than half the words in the evaluation corpus have been encountered before.[6]

[6] The reason the baseline does not show 100% pre-

6.1 Previous results

The results for the first two approaches were published in detail in a previous paper (Hulden et al., 2011).

6.1.1 Results using lexdiff

After experiments we may note that applying more than one rule within a word has a negative effect on the precision while not substantially improving the recall. Applying the unigram filter—choosing the most frequent candidate—yields a significant improvement: much better precision but also slightly worse recall. Choosing either parallel or sequential application of rules (when more than one rule is applied to a word) does not change the results significantly. Finally, compacting the rules and producing context-sensitive ones is clearly the best option.

In all cases the F_1-score improves if the unigram filter is applied; sometimes significantly and sometimes only slightly. The best result is shown in table 2. The options to obtain this result are: frequency 2; 2 rules applied; in parallel; with contextual conditioning.

6.1.2 Results using ILP

The only variable parameter with the ILP method dictates how many times a word-pair must be seen to be used as a learning evidence for creating a replacement rule. As expected, the strongest result is obtained by using all word-pairs, i.e. setting the threshold to 1. This is the result shown in table 2.

Interestingly, adding the unigram filter that improved results markedly in method 1 to the output of the ILP method slightly worsens the results in most cases, and gives no discernible advantage in others. In other words, in those cases where the method provides multiple outputs, choosing the most frequent one on a unigram frequency basis gives no improvement over not doing so.

6.2 Results using WFST

The experiments done by cross-validation with the development corpus to decide the best values of the parameters for the test have consisted in increasing the number of retrieved answers (1, 3, 5, 10, 20 or 30) and varying the search beam (default value or 5,000).

In all the WFSTs, specifying a search beam of 5,000 is better than using the default beam. As regards the number of answers, retrieving more answers yields a better F_1-score in the three WFSTs until an upper limit is reached. The upper limit is reached at 20 answers using morpheme sequences (WFST2 and WFST3). In WFST1, the plateau is reached at 5. Table 3 shows the results obtained.

Another important conclusion is that managing this value N we can balance precision and recall.

Finally, table 4 shows the final results obtained using the 80% of the corpus to train and the 20% to test. As the table shows, the best results are obtained using the last WFST. The differences among them are not statistically significant (p-values > 0.1 using Bhapkar's test). Anyway identifying morphemes is interesting for our future work (learning paradigms).

These results are overall consistently better than the ones obtained with the previous methods (see table 2).

	P	**R**	**F_1**
Baseline	95.62	43.52	59.82
Lexdiff	75.10	60.13	66.79
ILP	85.02	58.47	69.29

Table 2: The best results (per F_1-score) obtained with the first two methods).

	P	**R**	**F_1**
WFST1 (N=5)	82.88	72.81	**77.50**
WFST2 (N=20)	85.27	73.70	**79.05**
WFST3 (N=20)	83.74	73.87	**78.48**

Table 3: Average results obtained by cross-validation on development corpus with the three WFSTs. N is the number of asked answers. WSFT1: word/word. WFST2: word/morph-seq. WFST3: word/simpl-morph-seq. In all the cases the search beam is 5000.

cision is that the corpus contains minor inconsistencies or accepted alternative spellings, and our method of measuring the precision suffers from such examples by providing both learned alternatives to a dialectal word, while only one is counted as being correct.

	P	**R**	**F_1**
WFST1	83.46	75.42	79.23
WFST2	85.39	75.75	80.28
WFST3	85.56	76.74	**80.91**

Table 4: Results obtained in the final test with three WFSTs.

7 Conclusions and future work

We have presented a number of experiments to solve a very concrete task: given a word in the Lapurdian dialect of Basque, produce the equivalent standard Basque word. As background knowledge, we have a complete standard Basque morphological analyzer and a limited parallel corpus of dialect and standard text. The approach has been based on the idea of extracting string-to-string transformation rules from the parallel corpus, and applying these rules to unseen words. We have been able to improve on the results of a naive baseline using three methods to infer phonological rules of the information extracted from the corpus and applying them with finite state transducers.

When weights have been inferred the results have been improved.

The results using noisy-channel model (implemented using the *Phonetisaurus* tool) and standard morphological analysis seems very promising. In order to improve on these results, we plan to study the combination of the previous methods with other ones which infer dialectal paradigms and relations between lemmas and morphemes for the dialect and the standard. These inferred relations could be contrasted with the information of a larger corpus of the dialect without using an additional parallel corpus.

Acknowledgments

We are in debt to Josef Novak for his useful help using Phonetisaurus. This research has been partially funded by the Spanish Science and Innovation Ministry (Tacardi project, TIN2012-38523-C02-01) and by the Basque Government (Ber2tek, Etortek-IE12-333).

References

Alegria, I., Aranzabe, M., Ezeiza, N., Ezeiza, A., and Urizar, R. (2002). Using finite state technology in natural language processing of basque. In *LNCS: Implementation and Application of Automata*, volume 2494, pages 1–12. Springer.

Almeida, J. J., Santos, A., and Simoes, A. (2010). Bigorna–a toolkit for orthography migration challenges. In *Seventh International Conference on Language Resources and Evaluation (LREC2010), Valletta, Malta*.

Beesley, K. R. and Karttunen, L. (2002). Finite-state morphology: Xerox tools and techniques. *Studies in Natural Language Processing. Cambridge University Press*.

Hulden, M. (2009). Foma: a finite-state compiler and library. In *Proc. of the 12th Conference of the EACL*, pages 29–32, Athens, Greece. ACL.

Hulden, M., Alegria, I., Etxeberria, I., and Maritxalar, M. (2011). Learning word-level dialectal variation as phonological replacement rules using a limited parallel corpus. In *Proc. of the Dialects'2011. EMNLP*, pages 39–48.

Kestemont, M., Daelemans, W., and Pauw, G. D. (2010). Weigh your words—memory-based lemmatization for Middle Dutch. *Literary and Linguistic Computing*, 25(3):287–301.

Koskenniemi, K. (1991). A discovery procedure for two-level phonology. *Computational Lexicology and Lexicography: A Special Issue Dedicated to Bernard Quemada*, pages 451–446.

Mann, G. S. and Yarowsky, D. (2001). Multipath translation lexicon induction via bridge languages. In *Proc. of the second meeting of the NAACL*, NAACL '01, pages 1–8. Association for Computational Linguistics.

Muggleton, S. and De Raedt, L. (1994). Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679.

Novak, J. R., Minematsu, N., and Hirose, K. (2012). WFST-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding. In *Proc. of the 10th FSMNLP*.

Scherrer, Y. (2007). Adaptive string distance measures for bilingual dialect lexicon induction. In *Proceedings of the 45th Annual Meeting of the ACL.*, ACL '07, pages 55–60. ACL.