Deciphering Related Languages

Nima Pourdamghani, Kevin Knight
Information Sciences Institute & Department of Computer Science
University of Southern California
{damghani,knight}@isi.edu

Abstract

We present a method for translating texts between close language pairs. The method does not require parallel data, and it does not require the languages to be written in the same script. We show results for six language pairs: Afrikaans/Dutch, Bosnian/Serbian, Danish/Swedish, Macedonian/Bulgarian, Malaysian/Indonesian, and Polish/Belorussian. We report BLEU scores showing our method to outperform others that do not use parallel data.

1 Introduction

Statistical Natural Language Processing (NLP) tools often need large amounts of training data in order to achieve good performance. This limits the use of current NLP tools to a few resource-rich languages. Assume an incident happens in an area with a low-resource language, known as the Incident Language (IL). For a quick response, we need to build NLP tools with available data, as finding or annotating new data is expensive and time consuming. For many languages this means that we only have a small amount of often out-of-domain parallel data (e.g. a Bible or Ubuntu manual), some monolingual data and almost no annotation such as part of speech tags.

Fortunately, many low-resource languages have one or more higher-resource, closely Related Languages (RL). Examples of such IL/RL pairs are Afrikaans/Dutch and Bosnian/Serbian. A natural idea is to use RL resources to improve the task for IL. But this requires some kind of conversion between RL and IL. Assume the required NLP capability is named entity tagging. If we can convert RL to IL, we can convert all RL training data along with annotations into IL and train the tagger for IL. Or, if we can convert IL to RL we can use the potentially existing RL named entity tagger on converted IL data and project back the tags.

Following this idea, Currey et al. (2016) use a rule-based translation system to convert Italian and Portuguese into Spanish, to improve Spanish (here, IL) language modeling, Nakov and Ng (2009) convert RL/English parallel data to IL/English where both RL and IL have Latin orthography to improve IL/English machine translation. Hana et al. (2006) use cognates to adapt Spanish resources to Brazilian Portuguese to train a part-of-speech tagger. Mann and Yarowsky (2001) use Spanish/Portuguese cognates to convert an English/Spanish lexicon to English/Portuguese. These works prove the usefulness of RL data to improve NLP for IL, but they are designed for specific tasks and IL/RL pairs.

In this paper we propose a universal method for translating texts between closely related languages. We assume that IL and RL are mostly cognates, having roughly the same word order. Our method is orthography-agnostic for alphabetic systems, and crucially, it does not need any parallel data. From now on, we talk about converting RL to IL, but the method does not distinguish between RL and IL; as mentioned above, each direction of translation can have its own potential uses.

To translate RL to IL, we train a character-based cipher model and connect it to a word-based language model. The cipher model is trained in a noisy channel model where a character language model produces IL characters and the model converts them to RL. Expectation Maximization is used to train the model parameters to maximize the likelihood of a set of RL monolingual data. At decoding time, the cipher model reads the RL text character by character in which words are separated by a special character, and produces a weighted lattice of characters representing all the possible translations for each of the input tokens.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2513–2518
Copenhagen, Denmark, September 7–11, 2017.

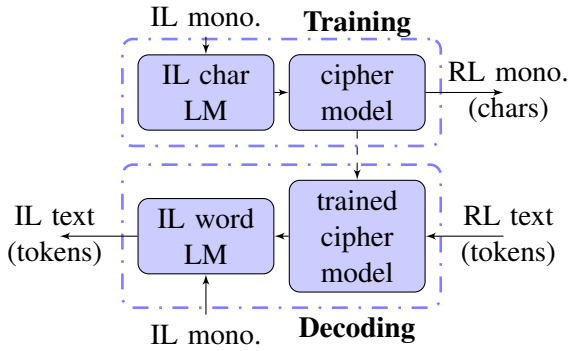

Figure 1: The process used for training the cipher model and decoding RL text to IL

The word-based language model takes this lattice and produces a sequence of output words that maximize the language model score times the cipher model score. Figure 1 depicts this process.

Our cipher models one-to-one, one-to-two and two-to-one character mappings. This allows us to handle cases like Cyrillic 'ч' and Latin 'ch', and also subtle differences in pronunciation between RL and IL like Portuguese 'justiça' and Spanish 'justicia'. Using a character-based cipher model provides the flexibility to generate unseen words. In other words, the vocabulary is limited by the decoding LM, not the cipher model. Separation of training and decoding language models enables us to train the decoding LM on as much data as is available without worrying about training speed or memory issues. We can also transliterate out of vocabulary words by spelling out the best path produced by cipher model in case no good match is found for a token in the decoding LM.

2 Related Work

Previous work on translation between related languages can be categorized into three groups:

Systems for specific language pairs such as Czech-Slovak (Haji et al., 2000), Turkish-Crimean Tatar (Cicekli, 2002), Irish-Scottish Gaelic (Scannell, 2006), and Indonesian-Malaysian (Larasati and Kubo, 2010). Another similar trend is translation between dialects of the same language like Arabic dialects to standard Arabic (Hitham et al., 2008; Sawaf, 2010; Salloum and Habash, 2010). Also, work has been done on translating back the Romanized version of languages like Greeklish to Greek (Chalamandaris et al., 2006) and Arabizi to Arabic (May et al., 2014). These methods cannot be applied to our problem because time and resources are limited to build a translation system for the specific language pair.

Machine learning systems that use parallel data: These methods cover a broader range of languages but require parallel text between related languages. They include character-level machine translation (Vilar et al., 2007; Tiedemann, 2009) or combination of word-level and character-level machine translation (Nakov and Tiedemann, 2012) between related languages.

Use of non-parallel data: Cognates can be extracted from monolingual data and used as a parallel lexicon (Hana et al., 2006; Mann and Yarowsky, 2001; Kondrak et al., 2003). However, our task is whole-text transformation, not just cognate extraction.

Unsupervised deciphering methods, which require no parallel data, have been used for bilingual lexicon extraction and machine translation. Word-based deciphering systems ignore sub-word similarities between related languages (Koehn and Knight, 2002; Ravi and Knight, 2011b; Nuhn et al., 2012; Dou and Knight, 2012; Ravi, 2013). Haghighi et al. (2008) and Naim and Gildea (2015) propose models that can use orthographic similarities. However, the model proposed by (Naim and Gildea, 2015) is only capable of producing a parallel lexicon and not translation. Furthermore, both systems require the languages to have the same orthography and their vocabulary is limited to what they see during training.

Character-based decipherment is the model we use for solving this problem. Character-based decipherment has been previously applied to problems like solving letter substitution ciphers (Knight et al., 2006; Ravi and Knight, 2011a) or transliterating Japanese katakana into English (Ravi and Knight, 2009), but not for translating full texts between related languages.

3 Translating RL to IL

We learn a character-based cipher model for translating RL to IL. At decoding, this model is combined with a word based IL language model to produce IL text from RL.

3.1 Cipher Model

Our noisy-channel cipher model converts a sequence of IL characters $s_1, ..., s_n$ to a sequence of RL characters $t_1, ..., t_m$. It is a WFST composed of three components (Figure 2):

WFST1 is a one-to-one letter substitution model. For each IL character s it writes one RL character t with probability $p_1(t|s)$.

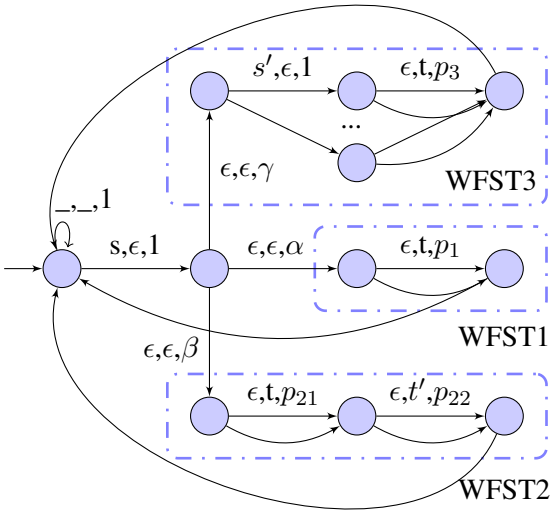

Figure 2: Part of the cipher model corresponding to reading IL character s from start state. The same pattern repeats for any IL character. After reading s, the model goes to WFST1, WFST2, or WFST3 with respective probability $\alpha(s)$, $\beta(s)$, or $\gamma(s)$. In WFST1, the model produces each RL character t with probability $p_1(t|s)$. In WFST2, the model produces each two RL characters t and t' with probability $p_{21}(t|s)$ and $p_{22}(t'|s)$. In WFST3, the model reads each IL character s' and produces each RL character t with probability $p_3(t|ss')$. From the last state of WFST1, WFST2, and WFST3, the model returns to the start state without reading or writing.The model has a loop on start state that reads and writes space.

WFST2 is a one-to-two letter substitution model. For each IL character s, it writes two RL characters t and t' with respective probabilities $p_{21}(t|s)$ and $p_{22}(t'|s)$.

We assume $p_{22}(t'|s)$ is independent of t. As a result we can estimate $p(tt'|s) = p(t|s)p(t'|t,s) \simeq p_{21}(t|s)p_{22}(t'|s)$ as modeled in WFST2. This simplification is required to make the model practicable. Otherwise, the size of the cipher model would become cubic in the number of RL and IL characters, and combining it with a language model would make the system unfeasibly large for training.

WFST3 is a two-to-one letter substitution cipher. For each IL character s, it reads another IL character s' with probability 1, and then writes one RL character t with probability $p_3(t|ss')$. As we will discuss in Section 3.2 we train p_3 directly from p_{21} and p_{22}, hence the cubic number of parameters does not cause a problem.

The start state reads each IL character s and goes to WFST1, WFST2, or WFST3 with respective probability $\alpha(s)$, $\beta(s)$, or $\gamma(s)$. The last state of each component returns to start without reading or writing anything. The start state also reads and writes space with probability one.

3.2 Training the Model

The cipher model described in Section 3.1 is much more flexible than a one-to-one letter substitution cipher. A few thousand sentences of RL monolingual data is not enough to train the model as a whole, and more training data makes the process too slow to be practical. Hence, we break the full model into WFST1, WFST2, and WFST3 and train the parameters of each component, i.e. p_1, p_{21} and p_{22}, and p_3 in separate steps. A final step trains the probability of moving into each of the components, i.e. α, β, and γ.

Each step of the training uses EM algorithm to maximize the likelihood of 500 sentences of RL text in a noisy channel model where a fixed 5-gram character based IL language model (trained on 5000 IL sentences) produces an IL text character by character and the cipher model converts RL characters into RL (top section of Figure 1).

Step one: We set $\alpha(s) = 1$ and $\beta(s) = \gamma(s) = 0$ and train $p_{1IL \to RL}(t|s)$ for each IL character s and each RL character t. In parallel we reverse RL and IL and train $p_{1RL \to IL}(s|t)$ for each RL character t and each IL character s. We use $p_1(t|s) = \frac{1}{2}(p_{1IL \to RL}(t|s) + p_{1RL \to IL}(s|t))$ to set WFST1 parameters in the next steps.

Step two: We set $\alpha(s) = \beta(s) = \frac{1}{2}$ and $\gamma(s) = 0$, fix p_1 and train $p_{21IL \to RL}(t|s)$ and $p_{22IL \to RL}(t'|s)$ for each IL character s and each pair of RL characters t and t'. In parallel we reverse RL and IL and train $p_{21RL \to IL}(s|t)$ and $p_{22RL \to IL}(s'|t)$ for each RL character t and each pair of IL characters s and s'.

Step three: Our cipher model has to decide after reading one IL character if it will perform a one-to-one, one-to-two or two-to-one mapping. In the first two scenarios the model has enough information to decide, but for the two-to-one mapping the model has to decide before reading the second IL character. For instance, consider converting Bosnian to Serbian. When the model reads the character "c" it has to decide between one-to-one, one-to-two and two-to-one mappings. A good decision will be two-to-one mapping because "ch" maps to ч, hence the system learns a large γ for character "c" but the same γ applies to any other

	afr	dut	bel	pol	bos	srb	dan	swe	mkd	bul	mal	ind
#sent.	0.9M	4M	1.9M	4M	0.8M	1.6M	4M	4M	0.9M	2.1M	0.4M	4M
#tok.	19M	74M	26M	58M	17M	29M	71M	70M	16M	33M	8M	75M

Table 1: Size of monolingual data available for each language. IL and RL are presented in pairs, IL first.

character that follows "c" which is not desirable.

One way to overcome this problem is to change the model to make the decision after reading two IL characters, but this will over-complicate the model. We use a simpler trick instead. We compute $p_{3IL \rightarrow RL}(t|ss')$ from $p_{21RL \rightarrow IL}(s|t)$ and $p_{22RL \rightarrow IL}(s'|t)$ using Bayes rule:

$$p_3(t|ss') = \frac{p(ss'|t)p(t)}{p(ss')} \simeq \frac{p_{21}(s|t)p_{22}(s'|t)p(t)}{p(ss')} \quad (1)$$

The estimate is based on our assumption from the previous step that $p_{22}(s'|t)$ is independent of s. For each RL character t we compute the empirical probability $p(t)$ from monolingual data and $p(ss')$ is the normalization factor.

We set p_3 parameters using equation (1), but before normalizing we manually prune the probabilities. If for IL characters s and s' there exists no RL character t such that $p_{21}(s|t)p_{22}(s'|t)p(t) > 0.01$ we assume that ss' does not map to any RL character. Otherwise, we only keep RL characters for which $p_{21}(s|t)p_{22}(s'|t)p(t) > 0.01$ and then apply the normalization.

Step four: In the final step we fix p_1, p_{21}, p_{22}, and p_3 to the trained values and train $\alpha(s)$, $\beta(s)$, and $\gamma(s)$ for each IL character s.

3.3 Decoding

In the decoding step we compose the cipher WFST with an IL word based language model WFST and find the best path for the input sentence in the resulting WFST (bottom section of Figure 1). If the best path has a high enough score the model outputs the corresponding IL token. Otherwise it outputs the highest scored character sequence produced by the cipher model as and OOV. In our experiments we use 1-gram and 2-gram language models trained on all the existing IL monolingual data (Table 1).

4 Data

We collect data for six pairs of related languages: Afrikaans(afr) / Dutch(dut), Bosnian(bos) / Serbian(srb), Danish(dan) / Swedish(swe), Macedonian(mkd) / Bulgarian(bul), Malaysian(mal) / Indonesian(ind), and Polish(pol) / Belorussian(bel). For each language, we download the monolingual data from Leipzig corpora (Goldhahn et al., 2012). The domain of the data is news, web, and Wikipedia. We consider the language with more data as RL and the one with less data as IL. Table 1 shows the size of available data for each language.

We also extract the list of alphabets for each language from Wikipedia, and collect the Universal Declaration of Human Rights (UDHR) for each IL and RL. We manually sentence align these documents and get 104 sentences and about 1.5K tokens per language. We use these documents for testing the conversion accuracy.

We tokenize and lowercase all the monolingual, parallel and UDHR data with Moses scripts. We remove all non-alphabetic characters from each text according to the alphabet extracted from Wikipedia. This includes numbers, punctuations, and rare/old characters that are not considered as official characters of the language. We keep all the accented variations of characters.

5 Experiments

We translate the UDHR between the related languages using the following methods:
Copy: Copying the text. This is not applicable for languages with different orthography.
LS: One-to-one Letter Substitution cipher. This is equivalent to using WFST1 without a decoding language model.
LS+1g LM: One-to-one letter substitution cipher with a 1-gram word language model at decoding.
PM+1g LM, **PM+2g LM:** The Proposed Method with respectively 1-gram and 2-gram word language model at decoding.

Results are reported for both directions of translation in Tables 2, and 3. For all the language pairs except Malaysian(mal) / Indonesian(ind), the proposed method is the best model with a large margin. Malaysian/Indonesian is a special case where, although the languages have a different vocabulary and a slightly different grammar, they have a common alphabet, and almost all of their cognates are exactly the same. See Figure 3 for an example. As a result the proposed method cannot learn much more than copying.

	afr→dut	bel→pol	bos→srb	dan→swe	mkd→bul	mal→ind
Copy	1.9 / 29.1	- / -	- / -	1.2 / 17.6	5.6 / 34.2	10.0 / 42.7
LS	1.9 / 29.1	0.0 / 7.9	33.2 / 59.4	1.3 / 20.7	9.1 / 39.1	10.0 / 42.7
LS+1g LM	2.9 / 33.3	0.7 / 15.1	32.8 / 59.2	4.5 / 31.3	9.1 / 39.1	10.3 / 43.1
PM+1g LM	4.1 / 36.3	0.0 / 21.8	39.9 / 64.6	6.4 / 36.5	11.8 / 43.3	10.4 / 42.8
PM+2g LM	4.3 / 36.2	1.9 / 24.2	39.2 / 64.4	6.9 / 38.8	11.9 / 43.0	10.4 / 42.8

Table 2: BLEU scores for IL-to-RL translation of UDHR text. Format is BLEU4/BLEU1. Polish / Belorussian and Serbian / Bosnian have different orthographies hence copying is not applicable.

	dut→afr	pol→bel	srb→bos	swe→dan	bul→mkd	ind→mal
Copy	1.9 / 25.0	- /-	- / -	1.2 / 18.7	5.6 / 33.5	10.0 / 41.6
LS	2.13 / 26.5	0.0 / 12.8	33.3 / 60.6	1.3 / 20.7	5.94 / 34.6	10.0 / 41.6
LS+1g LM	3.07 / 27.6	0.7 / 19.7	33.0 / 60.5	3.8 / 32.7	6.9 / 37.6	10.1 / 41.7
PM+1g LM	3.9 / 29.4	1.3 / 23.7	42.3 / 67.8	7.7 / 41.1	9.4 / 40.6	10.1 / 41.7
PM+2g LM	5.2 / 31.2	1.9 / 25.2	42.3 / 67.8	7.6 / 41.2	10.2 / 41.3	10.0 / 41.7

Table 3: BLEU scores for RL-to-IL translation of UDHR text. Format is BLEU4/BLEU1. Polish / Belorussian and Serbian / Bosnian have different orthographies hence copying is not applicable.

```
mal: semua manusia dilahirkan bebas   dan samarata dari segi kemuliaan dan hakhak
ind: semua orang   dilahirkan merdeka dan mempunyai martabat dan hakhak yang sama
```

Figure 3: First sentence of the first article of UDHR in Malaysian (mal) and Indonesian (ind). These languages have a different vocabulary, but their cognates (shown in bold) are exact matches.

```
afr:    alle menslike wesens word   vry   met  gelyke  -- waardigheid en  regte
a2d:    alle menslike wezens werd   vrij   met  gelijke -- waardigheid en  rechte
dut:    alle mensen   ------ worden vrij   en   gelijk  in waardigheid en  rechten
afr2en: all  human    beings are    free  with equal   -- dignity     and rights
a2d2en: all  human    beings were   free  with equal   -- dignity     and straight
dut2en: all  people   ------ are    free  and  equal   in dignity     and rights
```

Figure 4: First sentence of the first article of UDHR in Afrikaans (afr), Dutch (dut) and its conversion from Afrikaans to Dutch using PM+2-gram LM (a2d), along with their translations to English.

The proposed method translates between Serbian (srb) and Bosnian (bos) almost perfectly. For other pairs, we translate between a quarter and half of the words correctly, but we get few of the higher n-grams. Figure 4 visualizes the conversion of the first sentence of the first article of UDHR from Afrikaans (afr) to Dutch (dut) using PM+2g LM (4.3 BLEU4, 36.2 BLEU1). Observe that 4 out of 10 tokens are translated correctly, close to the 36.2 BLEU1 score, and there is no 3 or 4-gram match. For other tokens except "menslike" the translation is either correct but non-existent in the dutch sentence (wezens = beings, met = with) or has a meaning similar enough that can be useful in the downstream applications (werd = were v.s. worden = are, gelijke = equal(noun) v.s. gelijk = equal(adjective), rechte = straight/right v.s. rechten = rights). The token "menslike" in a2d is an OOV. The model is not able to convert "menslike" (afr) to "mensen" (dut). The language model does not accept other potential conversions and passes out "menslike" (a2d) as the best output of the cipher model.

6 Conclusion

In this paper we present a method for translating texts between closely related languages with potentially different orthography, without needing any parallel data. The only requirement is a few thousand lines of monolingual data for each language and a word language model for the target. Our experiments on six language pairs show the proposed method outperforms others that do not use parallel data.

Acknowledgments

This work was supported by DARPA contract HR0011-15-C-0115. The authors would like to thank Marjan Ghazvininejad, Ulf Hermjakob and Jonathan May for their comments and suggestions.

References

Aimilios Chalamandaris, Athanassios Protopapas, Pirros Tsiakoulis, and Spyros Raptis. 2006. All Greek to me! An automatic Greeklish to Greek transliteration system. In *Proc. LREC*.

Ilyas Cicekli. 2002. A machine translation system between a pair of closely related languages. In *Proc. ISCIS*.

Anna Currey, Alina Karakanta, and Jonathan Poitz. 2016. Using related languages to enhance statistical language models. In *Proc. NAACL*.

Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Proc. EMNLP*.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 Languages. In *Proc. LREC*.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Klein Dan. 2008. Learning bilingual lexicons from monolingual corpora. In *Proc. ACL*.

Jan Haji, Hric Jan, and Kubo Vladislav. 2000. Machine translation of very close languages. In *Proc. ANLP*.

Jirka Hana, Anna Feldman, Chris Brew, and Luiz Amaral. 2006. Tagging Portuguese with a Spanish tagger using cognates. In *Proc. ACL workshop on Cross-Language Knowledge Induction*.

Abo Bakr Hitham, Khaled Shaalan, and Ibrahim Ziedan. 2008. A hybrid approach for converting written Egyptian colloquial dialect into diacritized Arabic. In *Proc. INFOS*.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proc. COLING*.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proc. ACL workshop on Unsupervised lexical acquisition*.

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. In *Proc. NAACL*.

Septina Dian Larasati and Vladislav Kubo. 2010. A study of Indonesian-to-Malaysian MT system. In *Proc. MALINDO workshop*.

Gideon S Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Proc. NAACL*.

Jonathan May, Yassine Benjira, and Abdessamad Echihabi. 2014. An Arabizi-English social media statistical machine translation system. In *Proc. AMTA*.

Iftekhar Naim and Daniel Gildea. 2015. Feature-based decipherment for large vocabulary machine translation. In *arXiv*.

Preslav Nakov and Hwee Tou Ng. 2009. Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *Proc. EMNLP*.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proc. ACL*.

Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proc. ACL*.

Sujith Ravi. 2013. Scalable decipherment for machine translation via hash sampling. In *Proc. ACL*.

Sujith Ravi and Kevin Knight. 2009. Learning phoneme mappings for transliteration without parallel data. In *Proc. ACL*.

Sujith Ravi and Kevin Knight. 2011a. Bayesian inference for Zodiac and other homophonic ciphers. In *Proc. ACL*.

Sujith Ravi and Kevin Knight. 2011b. Deciphering foreign language. In *Proc. ACL*.

Wael Salloum and Nizar Habash. 2010. Dialectal to standard Arabic paraphrasing to improve Arabic-English statistical machine translation. In *Proc. ACL workshop on algorithms and resources for modeling of dialects and language varieties*.

Hassan Sawaf. 2010. Arabic dialect handling in hybrid machine translation. In *Proc. AMTA*.

Kevin P. Scannell. 2006. Machine translation for closely related language pairs. In *Proc. LREC Workshop on Strategies for developing machine translation for minority languages*.

Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In *Proc. EAMT*.

David Vilar, Jan-T. Peter, and Hermann Ney. 2007. Can we translate letters? In *Proc. ACL workshop on Statistical Machine Translation*.