CLASS-BASED NAMED ENTITY TRANSLATION IN A SPEECH TO SPEECH TRANSLATION SYSTEM

Sameer R. Maskey, Martin Čmejrek, Bowen Zhou, Yuqing Gao

smaskey@us.ibm.com, cmejrek@us.ibm.com
IBM T.J. Watson Research Center, Yorktown Heights, New York

ABSTRACT

Named Entity (NE) Translation is a challenging problem in Machine Translation (MT). Most of the training bi-text corpora for MT lack enough samples of NEs to cover the wide variety of contexts NEs can appear in. In this paper, we present a technique to translate NEs based on their NE types in addition to a phrase-based translation model. Our NE translation model is based on a syntax-based system similar to [1]; but we produce syntax-based rules with non-terminals as NE types instead of general non-terminals. Such class-based rules allow us to better generalize the context NEs. We show that our proposed method obtains a relative improvement of 1.41% in BLEU score over the baseline of phrase-based model in NE test set.

Index Terms— Named Entities, Machine Translation, Class-based Models

1. INTRODUCTION

The current data-driven statistical methods of MT such as IBM Models [2], phrase-based models [3], and syntax-based system [1] all require significant amount of data. The probability estimates for the translation get more robust as the counts of samples for each parameter increase. These parameters may model fertility, alignment, phrase segmentation, and other features of MT engine. The problem arises when there is not enough data to estimate the distribution of some of the parameters, particularly when we are trying to translate NEs. Let us describe a few problems that make the NE translation task difficult.

First, there are not enough instances of NE in the training data to cover the variety of contexts NE can appear in. A NE such as "John" can appear in many different contexts such as "I am John. John went home. She will call John. John and I are friends." MT model will have higher chances of error if the NE appears in context that has not been seen in the training data.

Second, it is never possible to cover all possible NEs that would exist in any unseen data set, so the out-of-vocabulary (OOV) rate is much higher for NEs than for any other class of words. OOV words are problematic for speech-to-speech translation system because ASR engine would not be able to recognize the words and MT engine would not be able to translate them. We have also seen in literature that OOVs amplify the word error rate by mis-recognizing the surrounding words as well.

In this paper, we try to address the first of the problems mentioned above, the problem of not having enough instances of NE appearing in various contexts. We propose a method based on classing of NEs that is similar to the class based language model concept, but we apply it the translation framework. We identify the NE types (PERSON or PLACE) and use them to produce syntax based rules. The rules contain non-terminals representing NE classes instead of undefined non-terminals as in [1].

We describe our syntax based NE translation system in Section 2. We then describe our two step process of automatically producing a new set of rules for NEs in Section 3. We present of results in Section 5 and conclusion in Section 6.

2. SYNXLATOR: AN ENGINE FOR NE TRANSLATION

In order to be able to better translate NEs, we built a translation engine, *synXlator*, which can better take account of contextual information of NEs. The *synXlator* is a hierarchical phrase-based machine translation model with a support for multiple classes, and with a decoder extended from a formally syntax-based SMT [4]. The extension that are made on a formally syntax-based SMT decoder like [1] are as follows: We allow a general non-terminal to be rewritten with recursive synchronous context-free grammar (SCFG). In addition, a number of class-specific non-terminals $Y_1, Y_2, \ldots Y_n$ are defined in the rules, with each representing one NE class. Similar to non-terminal X in [1], there must be a one-to-one mapping of Y in the source and target language part of the rule. However, these non-terminals are only required to be replaced once, i.e., no recursion is needed for them.

The extensions on decoder we described above allow us to generalize the context of NEs better. Let us look at an example of some rules that can be used in decoding of a sentence.

@NAME @NAME mdyr_bw mdrsp_bw || @1 @2 is a school principal || 0 0 0 0

@NAME @NAME mdrs_bw fyzyAU_bw || @1 @2 is a physics

teacher || 0 0 0 0
@NAME @NAME mdrs_bw || @1 @2 is a teacher || 0 0 0 0

In a standard phrase-based system like [3] the phrase pairs would contain phrases like "John Smith is a school principal" and a test sentence with a similar n-gram will get higher translation score. If "Benjamin Cole" appears instead of "John Smith" it may get a lower translation score since the translation engine has never seen such phrase pair. But our rules above generalize the context of "is a school principal" such that any name can appear to replace the non-terminal @1 and @2. The *synXlator* exploits this advantage by being able to take account of such generalization in a standard syntax-based decoding framework. We hope that our framework generalizes this small set to a much larger number of other appropriate NE entries that are suitable for the similar contexts. Furthermore, we note that we only need a small number of classes to represent different types of NE entries (e.g., PERSON and PLACE). Next, in the following section, we describe our two steps process to automatically produce these rules.

3. BUILDING NE CLASS-BASED TRANSLATION RULES

3.1. Step One

In the first step we extract rules from an English-Arabic parallel corpus with human annotated NEs. This corpus was provided as a part of TransTac June 2008 Evaluation. We make use of lists of NE pairs (Arabic name and its English translation/transliteration) for the following categories: MALE, FEMALE, PLACE, STREET, SURNAME, and TRIBE. From each sentence pair we want to extract rules by replacing some occurrences of NE pairs by appropriate non-terminals identifying the original NE category. The number of possible rules can be potentially huge, since a single NE pair may belong to multiple categories, a NE can span one or more words, as well as the number of replaced NE pairs in the sentence can vary from one to all.

In order to produce these rules efficiently, we built an algorithm, which we describe in Figure 1. Let C_k be a list of NE translation pairs for category c_k for $k = 0, \ldots, K$. Each NE consists of 1 or more words. We start from an initial hypothesis $\langle \mathbf{e}, \mathbf{f}, E, F, k = 0 \rangle$, where $\mathbf{e} = e_1, \ldots, e_m$ and $\mathbf{f} = f_1, \ldots, f_n$ are the English-Arabic sentence pair, $E : E \subseteq \{1, \ldots, m\}$ and $F : F \subseteq \{1, \ldots, n\}$ are sets storing indices of English and Arabic words marked as proper names, and finally, k is the maximum index of replaced category c_k for $k = 1, \ldots, K$. When extending a hypothesis, we try to replace one NE pair by appropriate non-terminal. In Step 5, we only iterate categories that have higher index than $h.k$ to exclude repetitions. Similarly, the first two conditions in Step 6 prevent from replacing NE occurrences on positions that were once skipped.

Since our training data is not perfect, incomplete translations are relatively frequent. To fix the most harmful errors,

1. initialize stack S
2. S.push($\langle \mathbf{e}, \mathbf{f}, E, F, 0 \rangle$)
3. **while** not emty(S)
4. **let** h = S.pop()
5. **for** each $k \in \{h.k, \ldots, K\}$
6. **for** each $\langle i, i', j, j' \rangle$ such that
   $\forall i^*, i' < i^* \leq m \Rightarrow h.e_{i^*} \neq @C_k$
   $\forall j^*, j' < j^* \leq n \Rightarrow h.f_{j^*} \neq @C_k$
   $i \in h.E$ and $j \in h.F$
   and $\langle h.e_i \ldots h.e_{i'}, h.f_j \ldots h.f_{j'} \rangle \in C_k$
7. **let** h' = copy(h)
8. replace $h'.e_i \ldots h.e_{i'}$ by $@C_k$
9. replace $h'.f_j \ldots h.f_{j'}$ by $@C_k$
10. **let** $h'.E = h.E \setminus i$
11. **let** $h'.F = h.F \setminus j$
12. output rule($\langle h'.e, h'.f \rangle$)
13. S.push(h')

Fig. 1. Algorithm generating NE translation rules

	corpus name	#pairs
1	standard training set	1,097,783
2	names_long	33,208
3	names_male	2,316
4	names_places	1,408
5	names_tribes	1,260
6	names_female	1,153
7	names_streets	664
8	names_surnames	848
9	dev_set	1,979
10	names_test_set	3,023

Table 1. Data sizes

we allow only such rules that have at least one terminal on both sides. Some of the rules generated by these steps are shown in Section 2. We can see from the rules that our non-terminals @NAME can take any name, hence each context NE appears in our training data has a growth factor of N where N is total NEs of a given class in the training data. Essentially, we allow any name in the given class to appear in the context provided by each rule.

3.2. Step Two

We extracted rules in Step One from sentences that contain NEs. In order to add more rules that will increase the context of words NEs appear in, we want to extract rules from phrase pairs as well. Obtaining rules from phrase pairs has two problems though, first, we need to identify the phrases to be used to extract the rules, and second, we need to know which phrase pairs to trust. Unlike sentence pairs, phrase pairs are generated with automatic phrase segmentation and alignment, hence the data is noisy. We need to exclude phrase pairs that may potentially generate rules with incorrect translations.

In order to identify phrase pairs from which we can gen-

C(i,j)	C(i) - C(i,j)	C(i)
C(j) - C(i,j)	N - C(i) - C(j) + C(i,j)	N - C(i)
C(j)	N - C(j)	N

Table 2. Contingency Table to test our Null Hypothesis

erate rules that will potentially improve the system, we start with Step One rules. We remove all the stop words from the Step One rules and rank the remaining words according to its frequency. Then we score the sentences according to the number of these relevant words occurring in the phrase pair. We weight the scores according to the positions of the relevant words in the frequency ranking. We combine this score with a pattern matching scores where patterns such as "* name is *" are used. If a sentence contains a matching pattern the score is incremented by 1. These patterns were generated manually by looking a the rules from Step One. The final combined score based on frequency and patterns tells us if we should use the given phrase pair to extract rules. Next we filter the phrase pairs by removing phrase pairs that potentially may have incorrect translations. We do such filtering using a procedure similar to [5].

The idea behind [5] is that we can throw away many phrase pairs that do not have sufficient statistics to back up their pairing. That is if a phrase pair e_i, f_j occurs only a few times but e_i pairs with other phrase pairs many times such that probability of e_i, f_j pairing together is no more than chance we can throw them away. We take a similar concept but instead of throwing away phrase pairs that are below the commonly used significance level of 0.05 we use a much higher significance level.

Our *null hypothesis* H_o is that the phrase pair e_i, f_j should not be paired together. In order to test our hypothesis for each phrase pair, we first build a contingency table that counts the number of times phrase pairs occurred together and the number of sentences the individual phrases occur. Our contingency table is shown in Table 2.

In the contingency table 2 the last column and the last row are marginals while N is the grand total, i.e. the count of all the phrase pairs. $C(i, j)$ is the count of total pair of sentences that had phrase e_i in the source language sentence and f_j in the target language sentence. $C(i) - C(i, j)$ tell us the number of sentences that had e_i in source language but did not contain f_j in the target language sentence. The above table can be used to compute the statistical significance of the relationship between them if the table entries represent a random sample of distribution used in null hypothesis. We can use any of significance testing methods such as Pearson's Chi Square test, G-test or Fisher's test. Even though [5] argues that Fisher's test is better for random variables representing counts in phrase pairs, in our experiment Pearson's test performed reasonably well. Hence, we perform a Pearson's test to compute the significance of the relationship between two variables. The value of degree of freedom for our 2x2 contingency table is 1.

We used a very strict significance level. We rejected the null hypothesis if the significance level provided by the test was below 0.0005. We kept all the phrase pairs for which we rejected the null hypothesis and threw the rest. Hence, after the significance testing we had phrase pairs for which the association between the phrase pairs was very strong according to their counts in the training data. Hence, the final set of selected phrase pairs had a very high likelihood of containing NEs, and had a very high probability of being correct translations.

We had a list of names and their translations from Step One. We replaced the words in phrase pairs with non-terminals if they were in our name list obtained from Step One. We only replaced the names when the matching translated name also appeared in foreign phrase. After we replaced the names with the non-terminals we had a final set of phrase pairs that had non-terminals representing the type of NE it can be replaced with. And we had a set of possible name translations for each class. We use this set as a set of rules of our syntax-based system.

We next describe our experiment with these rules and compare its performance with a phrase-based translation model.

4. EXPERIMENTS

We first built our baseline system and computed its performance. Second, we added the NE-class-based rule system to our baseline system and measured its performance. We tested the performance on a name heavy set and a large general test sets. These test sets are a subset of standard test sets provided for TransTac.

Our baseline system is a phrase-based translation model (PB) similar to [3] with a stack decoder. Our NE-class-based model is a type of syntax-based model as described in Section 2, with two different NE classes: NAME and PLACE. We want our NE-class-based model to have a very high precision. If the NE model is not able to translate using the given set of rules, it outputs an emty string. In such case, the combination system will default to the translation proposed by the baseline system.

For our baseline system, we trained Iraqi to English model with a bi-text corpus with approximately 100K training sentences. From this we had a held out test set and dev set for tuning. We built a tri-gram language model for English and trained a phrase-based model. We then tuned our model using the dev set. Then we tested the baseline model with one reference of a name heavy test set, ArabicNames. The results are shown in Table 3.

After we tested our baseline model, we tested the same test set, ArabicNames, with *synXlator* using the rules generated in Step One and Two. The *synXlator* returns empty string when the sentence cannot be translated using any of the rules. For such sentences we use the baseline models output. Let us define the set of sentences translated by the NE model as

Test Set	PB	PB +Step1	PB +Step1 +Step2	PB +Step1 +Step2 +merge
TestSet	BLEU			
ArabicNames	0.4606	0.4659	0.4672	0.4630
Jan07Live	0.5526	0.5523	0.5523	0.5509
Jan07Offline	0.6839	0.6834	0.6824	0.6822
TransTacPhase2	0.3902	0.3900	0.3899	0.3897
TransTacPhase3_34	0.2842	0.2841	0.2834	0.2832
NE freq. range	F_1-measure on ArabicNames			
1–5	0.5154	0.5229	0.5242	0.5134
6–20	0.8856	0.8921	0.8891	0.8838
21–100	0.8966	0.9037	0.9044	0.9032
>100	0.9462	0.9460	0.9464	0.9481
TOTAL	0.8683	0.8718	0.8709	0.8674

Table 3. Results for NE Translation Comparing to baseline Phrase-Based (PB) model

set A and the ones that could not be translated as B. Sentences of set B are translated by baseline model and final set of translated sentences is obtained by combining sentences of sets A and B. We then computed the BLEU [6] scores on this set of translated sentences. We experimented with NE-based models based on Step One only and both steps combined.

In order to make sure that our NE model is not harming the performance of the overall system, we also tested the class-based model on other general test sets. We tested its performance on the following test sets: Jan07Live, Jan07Offline, TransTacPhase2Dev, TransTacPhase3_34. These test set are subset of sentences provided in the standard sets from Darpa for TransTac 2008 Evaluation.

5. RESULTS AND DISCUSSION

The results in Table 3 show that our class-based NE model improves the baseline phrase-based approach by 1.41% of relative improvement in BLEU score, as well as by 0.26% absolute gain in F_1-measure for translated NEs (0.75% for the least frequent ones). We also tested how much the NE-based model degrades the overall performance on the other diverse sets. Even though a slight degradation in overall performance can be seen from Table 3, it is still less than 0.02% in all test cases, and significantly less than the improvement on the name set.

We also experimented with merging of PERSON and PLACE class into one NAME class to see if we can generalize better with merged classes. From the experiments we see that in all of the test sets merging classes degraded the performance. This shows that the information retained in independent classes of PERSON and PLACE is useful for overall NE translation. Also, this follows the intuition that many personal names are not used as place names.

Another observation we make from the results is that Step One rules tend to degrade less in other test sets than using Step One and Step Two rules combined. We think this is due to the nature of rules generated in these steps. In Step One, rules are created only when a sentence pair has NE. Thus the rules are fired only when exact phrase pair is found in a sentence. On other hand in Step Two we have rules that may not have NEs in them but have words that are very closely related with NE such as "name is" "live in." Such rules allow the system to use more combinations of rules during decoding even though there may not be exact match. The flexibility adds some noise but improves the coverage.

6. CONCLUSION

We presented a method for improving NE translation by building a class-based NE translation model. Two classes were added as non-terminals replacing the names in phrase pairs, creating a rule set extending a formally syntax-based system. The NE model worked in combination with phrase-based model to produce a better NE translation improving the baseline by relative 1.41% in BLEU score, as well as by 0.26% absolute of F_1-measure. Our experiments also showed that distinguishing between two different NE categories such as person and place using two different classes produced slightly better results than if both categories were merged.

7. ACKNOWLEDGMENT

This work is in part supported by the US DARPA under the TransTac program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

8. REFERENCES

[1] David Chiang, "A hierarchical phrase-based model for statistical machine translation," in *ACL*, 2005, pp. 263–270.

[2] Peter E Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer, "The mathematics of statistical machine translation: parameter estimation," *Computational Linguistics*, vol. 19, pp. 263–311, 1993.

[3] Franz Josef Och and Daniel Marcu, "Statistical phrase-based translation," in *HLT*, 2003, pp. 127–133.

[4] Bowen Zhou, Bing Xiang, Xiaodan Zhu, and Yuqing Gao, "Prior derivation models for formally syntax-based translation using linguistically syntactic parsing and tree kernels," in *ACL*, 2008.

[5] J Howard Johnson and Joel Martin, "Improving translation quality by discarding most of the phrasetable," in *EMNLP-CoNLL*, 2007.

[6] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu, "Bleu: A method for automatic evaluation of machine translation," in *ACL*, 2002, pp. 311–318.