Automatic Determination of Number of clusters for creating templates in Example-Based Machine Translation

Rashmi Gangadharaiah rgangadh@cs.cmu.edu
Ralf D. Brown ralf@cs.cmu.edu
Jaime Carbonell jgc@cs.cmu.edu

Language Technologies Institute,
Carnegie Mellon University,
Pittsburgh, PA 15213.

Abstract

Example-Based Machine Translation (EBMT), like other corpus based methods, requires substantial parallel training data. One way to reduce data requirements and improve translation quality is to generalize parts of the parallel corpus into translation templates. This automated generalization process requires clustering. In most clustering approaches the optimal number of clusters (N) is found empirically on a tune set which often takes several days. This paper introduces a spectral clustering framework that automatically estimates the optimal N and removes unstable oscillating points. The new framework produces significant improvements in low-resource EBMT settings for English-to-French ($\approx$1.4 BLEU points), English-to-Chinese ($\approx$1 BLEU point), and English-to-Haitian ($\approx$2 BLEU points). The translation quality with templates created using automatically and empirically found best N were almost the same. By discarding "incoherent" points, a further boost in translation scores is observed, even above the empirically found N.

1 Introduction

An EBMT system uses a parallel corpus to translate new input source sentences. In the Translation Model (TM), the input sentence to be translated is matched against the source sentences. When a match is found, the corresponding translation in the target language is obtained through sub-sentential alignment. In our EBMT system, the final target translation is obtained from these partial target translations with a beam-search decoder using a target Language Model (LM). EBMT systems require large amounts of data to function well (Brown, 2000).

Generalization using equivalence classes (Veale and Way, 1997; Brown, 2000) reduces the amount of pre-translated text required and improves translation quality. Translation templates (or short reusable sequences) are generalizations of source and target sentences where sequences of one or more words are replaced by variables. Various methods have been proposed to create such templates in EBMT and differ in the way the templates are created. Some systems required a full template to match the input source sentence for target sentence generation (Cicekli and Güvenir, 2001), others adopted a fragmentary translation algorithm (Kaji et al., 1992) where the target sentence generation is similar to that adopted in Transfer-based MT systems. Somers (1994) suggests using 'hooks' using alignment information and Block (2000) uses a simple translation algorithm that can join only single variable target fragments. Gaijin (Veale and Way, 1997) performs phrase chunking based on the marker hypothesis.

Templates resemble transfer rules used in Rule-Based MT (Lavie, 2008) but have fewer constraints. Syntax-based SMT (Yamada and Knight, 2001) also uses transfer rules on parsed input that contain only non-terminal symbols and use lexical transfer rules for translating the source words. In EBMT, lexical transfer rules consist only of terminal symbols and a generalization template contains non-terminal and terminal symbols. Templates provide a way to reorder target phrasal matches that are not necessarily linguistic phrases and a way to increase coverage. Candidates for template generation are POS tags or automatically clustered words. This paper adopts a simpler method to create templates for EBMT (similar to (Brown, 2000)) where automatically clustered word-pairs are replaced by their class labels.

Data sparsity has remained a great challenge even in statistical language modeling and templates can be used here to provide better probability estimates. Class-based models make reasonable predictions for unseen histories by using the class information of all the words present in the histories. Hence, class-based models require all words present in the training data to be clustered. When hand-made clusters are not available, automatic clustering tools can be used to obtain clusters. We modify the class-based LM, where only a small set of words are clustered to create templates and call it the template-based LM. It should be noted that the template-based model is equivalent to a class-based model formed by placing each of the words that were not clustered for building the template-model in a unique class, leading to singleton clusters for unclustered words. The model is similar to the factored language models (FLMs) (Kirchhoff and Yang, 2005), where a word can be represented by features. This results in a model space that is exponentially large. Our model is a much simpler version of FLMs where we use one extra feature other than the word itself and the backoff procedure adopted is fixed and not learnt.

Clustering algorithms can be used to group words (data points) based on the context in which they appear. Among these, spectral clustering algorithms (well known for their ability to identify non convex clusters) lead to intuitively pure clusters and higher translation scores (Gangadharaiah et al., 2006). These algorithms use the eigenstructure of a similarity matrix to partition data points into clusters. Although spectral clustering algorithms are powerful in forming pure clusters, in most applications, the number of clusters (N) is set manually. Parameters of MT systems are tuned based on the performance of the system on a development set, and must be retuned for each value of N. This is computationally expensive as the process can take several days. Ideally, one would like to use an algorithm that is simple in design, produces pure clusters and automatically finds N.

If all the data points in different clusters were infinitely far apart then one could easily find N for the spectral clustering algorithm by counting the number of eigenvalues that are equal to 1. However, clusters are not far apart in real world problems. An algorithm to automatically determine N was proposed by Sanguinetti et al. (2005) and tested on artificially constructed images. This method could not be applied directly to our EBMT system (Section. 4.1). We hypothesize that this is because of the noisy and imperfect nature of real data. This paper provides a solution: modify the algorithm to detect and remove outliers. We believe that these problems could arise in other practical systems and our modified algorithm would apply to those problems as well.

In essence, this work addresses the question of how to automatically generate clusters that contain mostly reliable words when hand-made clusters are not available to generate templates. The contribution of this paper is three-fold. First, an algorithm is proposed to automatically find the optimum N on real data (Section. 4.2). Second, we detect incoherent points (that do not fit in any cluster) and show how the performance improves by removing these points. Finally, we show an increase in translation quality (Section. 6) in sparse data conditions by creating generalized templates both in the TM (Section. 2) as well as in the LM (Section. 3) of an EBMT system.

2 Generalization in TM

We first motivate the use of templates in the TM. Assume the training corpus consists of just the following two sentence-pairs[1]. A single template T is formed by replacing, "Minister - *ministre*"/"President - *président*" and "Wednesday - *mercredi*"/"Monday - *lundi*" by their class names, <CL0> and <CL1> respectively, and indexed. Say input $\underline{I}$ needs to be translated.

Example training corpus:
S_1:The Minister gave a speech on Wednesday .
T_1:*Le ministre a donné un discours mercredi .*
S_2:The President gave a speech on Monday .
T_2:*Le président a donné un discours lundi .*

Example word-pair Clusters:[2]
<CL0>: Minister-*ministre*,President-*président*,..
<CL1>: Wednesday-*mercredi*,Monday-*lundi*,..

Generalized template (T):
The <CL0> gave a speech on <CL1> .
Le <CL0> a donné un discours <CL1> .

$\underline{I}$:The President gave a speech on Wednesday .

If no templates are used, then the TM would generate the following two phrasal matches from the corpus and place them on a common lattice:"The President gave a speech on"–*"Le président a donné un discours"* and "Wednesday"–*"mercredi"*. If a statistical decoder that uses a target LM is used, then the phrasal matches on the

[1]sentence-pair: source and its corresponding target sentence
[2]word-pair: source and its corresponding target word

lattice can be reordered to generate the output.

Many of the EBMT systems do not use a decoder and depend on the templates to combine and produce the output. In such systems, the input is converted to its template form (ITS) by replacing the words in the input sentence by variables (if the words belong to an equivalence class) and their translations - "<CL0> *président*" and "<CL1> *mercredi*" - are stored. If a matching template is not found then the sentence cannot be translated.

$\underline{ITS}$: The <CL0> gave a speech on <CL1>

The TM looks for matches in the indexed corpus. ITS completely matches T and hence the target template that corresponds to the matched source template in the corpus is obtained as a candidate template (ITT) for the input sentence.

$\underline{ITT}$: *Le <CL0> a donné un discours <CL1>*

The translations that were stored are put back into the template to obtain the output(O).

$\underline{O}$:*Le président a donné un discours mercredi*

Templates are also useful in EBMT systems that use statistical decoders. Present decoders have constraints on the amount they can reorder the target phrasal matches as it is computationally expensive to try all possible reorderings. For language pairs that have very different word order, it is better to extract longer phrasal matches from the TM. Templates provide a way to generate longer target phrasal matches. As seen above, ITT was obtained from the TM and the variables were replaced by the translations of the generalized words to produce a longer target phrasal match. [(*"Le président a donné un discours mercredi"*) vs. (*"Le président a donné"* and *"mercredi"*)].

2.1 Term Vectors

Using a bilingual dictionary created using statistical methods (Brown et al., 1990) and parallel text, a rough mapping between source and target words is created. When there is only a single possible translation listed for a word by the mapping, a word pair made up of the word and its translation is created. For each such word pair we then accumulate counts for each token in the surrounding context of its occurrences (n words, currently 3, immediately prior to and n words immediately following). These counts form a pseudo-document for each pair, which are then converted into term vectors for clustering as in Brown (2000).

3 Generalization in LM

Clustered words are replaced by their labels in the target language corpus to obtain templates. Suppose the target language corpus contains the sentences, S1 and S2. Say, *<ORG>* and *<WEEKDAY>* are example clusters. In templates, T1 and T2, *"school"* and *"office"* are replaced by *<ORG>* and *"Monday"* by *<WEEKDAY>*.

Target Corpus:
S1: *the school reopens on Monday .*
S2: *the office is too far .*

Example Clusters:
<ORG>: school, company, office
<WEEKDAY>: Monday, Tuesday, Wednesday,...

Templates:
T1: *the <ORG> reopens on <WEEKDAY> .*
T2: *the <ORG> is too far .*

Reusable templates of the above form are used to build the target LM. The process involved in building these models is similar to that of building word-based LMs except that now the conditional probability is based not just on words in the history but on class labels as well.

With a word-based LM (for simplicity, trained on the corpus containing only S1 and S2), if a subsequence such as *"the office reopens"* was encountered, the model would return less reliable scores for *p(reopens|the office)* by backing off to the unigram score, *p(reopens)*. However, the template-based model makes use of the available data well by converting the subsequence, *"the office reopens"* to *"the <ORG> reopens"* and hence, a more reliable score i.e., *p(reopens|the <ORG>)* contributes to the LM score of this sequence.

3.1 Template-based LM Formulation

A template-based LM can be given by eqn (1),

$$p(w_i|h) \approx xp(f_i|f_{i-1}, ..., f_{i-n+1}) \quad (1)$$

$$where f_j = \begin{cases} c(w_j), & \text{if } w_j^{th} \text{ class is present} \\ w_j, & \text{otherwise} \end{cases} \quad (2)$$

$$x = \begin{cases} p(w_i|c(w_i)), & \text{if } w_i^{th} \text{ class is present} \\ 1, & \text{otherwise} \end{cases} \quad (3)$$

The probability of the i^{th} word (w_i) given its history h is represented as the probability of feature f_i corresponding to w_i given its previous history of features. Each feature can represent a word, w_j or its class, $c(w_j)$ if w_j is clustered.

3.2 Incorporating Template-Based LM

The decoder works on a lattice of possible phrasal target translations (or fragments) for source phrases present in the input sentence to generate the target translation. Similar to Pharaoh (Koehn et al., 2003), the decoder uses multi-level beam search with a priority queue formed based on the number of source words translated. Bonuses are given for paths that have overlapping fragments (Brown, 2003). The total score (TS) for a path (eqn 4) through the translation lattice is the arithmetic average of the scores for each target word in the path. The EBMT engine assigns each candidate phrasal translation a quality score computed as a log-linear combination of alignment score and translation probability. The alignment score indicates the engine's confidence that the right target translation has been chosen for a source phrase. The translation probability is the proportion of times each distinct alternative translation was encountered out of all the translations.

$$TS = \frac{1}{n}\sum_{i=1}^{n}(wt_1 * log(b_i) + wt_2 * log(pen_i) \\ + wt_3 * log(q_i) + wt_4 * logP(w_i|w_{i-2}, w_{i-1}) \quad (4)$$

where, n is the number of target words in the path, wt_j indicates the importance of each score, b_i is the bonus factor given for long phrasal matches, pen_i is the penalty factor for source and target phrasal-length mismatches, q_i is the quality score and $P(w_i|w_{i-2}, w_{i-1})$ is the LM score.

The template LM builder takes in training data and a class file consisting of words with their corresponding equivalence classes. The model is built by replacing the words that occur in the class file by their class names. It should be noted that this model allows us to use only the reliable words to be replaced by their class names. The words and their class names are stored for future look ups for generalizing target fragments during decoding.

For the LM scores, words on the lattice are replaced by their equivalence classes and their n-gram probabilities are determined using the template-based LM. These scores are then interpolated with the probabilities obtained with the word-based model (eqn 5). Hill-climbing can be used to find the best λ on a tuning set.

$$\begin{aligned} p(w_i|h) &= \lambda[xp(f_i|f_{i-1}, ..., f_{i-n+1})] + \\ &\quad (1-\lambda)p(w_i|w_{i-1}, .., w_{i-n+1}) \quad (5) \end{aligned}$$

4 Automatic determination of number of clusters

The algorithm described in (Gangadharaiah et al., 2006) is used to cluster word pairs. The method uses the NJW algorithm (Ng. et. al., 2001) with certain variations proposed by Zelnik-Manor and Perona (2004) to compute local scaling parameters automatically for the affinity matrix and by Verma and Meila (2003) for the $k-$means orthogonal treatment during the initialization.

An algorithm was proposed by Sanguinetti et al. (2005) to automatically find the number of clusters (N). The intuition behind the method is as follows. When the rows of the k eigenvectors are clustered along mutually orthogonal vectors, their projections will cluster along radial directions in a lower dimensional space. When q is less than the best number of clusters (N), meaning that $[N-q]$ eigenvectors are discarded, the points that are not close to any of the first q centres get assigned to the origin. Elongated k-means is initialized with $q = 2$ centres and the $(q+1)^{th}$ centre as the origin. Their elongated k-means algorithm downweights distances along the radial direction and penalizes distances along the traversal direction. If points get assigned to the centre which originated from the origin, the value of q is incremented and the procedure is repeated. The procedure is terminated when no points get assigned to the $(q+1)^{th}$ centre.

4.1 Problems encountered

To see the performance of the algorithm on different language pairs, we separately applied the clustering algorithm and the resulting templates (i) to the TM of an English-French (Eng-Fre) EBMT system and (ii) to the LM of an English-Chinese (Eng-Chi) EBMT system. The analysis on 10k Eng-Fre is as follows. As seen in Fig. 1, the number of points assigned to the origin reaches zero in the 34^{th} iteration (when the number of clusters is 36). Hence, generalized templates were obtained with 35 clusters. These templates were used to translate test data in 10 test files (see Section 5 for details on test data). With experiments performed using generalized templates obtained with 35 clusters, the average BLEU score (Papineni et al., 2002) was found to be much less (difference of 1.3 BLEU points on average) than the BLEU scores with generalized templates obtained using N that was set experimentally (Table 1). The automatically determined N was not the same as the

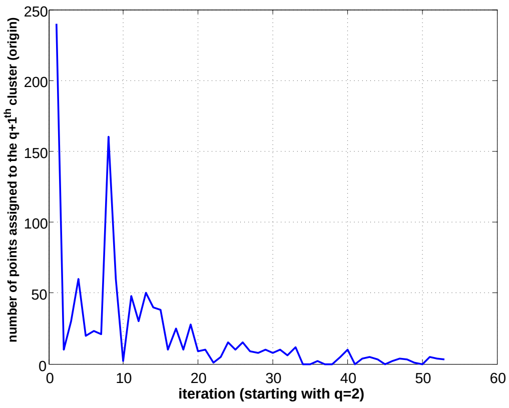

Figure 1: Plot of number of data points assigned to the origin in every iteration using (Sanguinetti et al., 2005) in EBMT for 10k Eng-Fre.

experimentally determined N. The same behavior was also found in the 30k Eng-Chi. To study the nature of the problem, the value of q was increased beyond 3 for the artificial image data consisting of 3 circles in Sanguinetti et al. (2005) and the number of points assigned to the origin was analyzed. When the value of q was increased beyond 3, the number of points assigned to the origin remained at zero. However, for real data in our case, as the value of q was increased beyond 34, fluctuations in number of points assigned to the origin were observed. Intuitively, these fluctuations could be due to the presence of data points that are hard to classify. They could also be treated as incoherent points or points in reality that do not belong to any cluster. The algorithm given in Section 4.2 removes these unclassifiable points from the rows of the U matrix (containing eigenvectors with greatest eigenvalues stacked in columns) and reruns the procedure to determine the optimum N.

4.2 Modified Algorithm

The algorithm starts with q=2 centres and $(q+1)^{th}$ centre as the origin. BP holds the first iteration number at which the number of points assigned to the origin was 0. it holds the number of consecutive iterations for which the number of points assigned to the origin was 0. At the start of the algorithm, BP is empty and it is 0. Elongated k-means is performed with q+1 centres. If there are points assigned to the $(q+1)^{th}$ centre, the value of q is incremented as in Sanguinetti et al. (2005). Say for the first time at iteration i there are no points assigned to the origin, then BP is set to i. If BP has been set and the number of points assigned to the origin is greater than 0 in the following iteration, then the points assigned to the origin are removed from the U matrix and the algorithm is rerun starting with q=2 centres. If the number of points assigned to the origin remains at 0 for 4 consecutive iterations (Fig. 2), the procedure is terminated and the best N is given by BP-4. We confirmed experimentally that if there were no points assigned to the origin for 4 consecutive iterations, then there were no points assigned to the origin in the future iterations. Table 1 shows the average BLEU score obtained on 10k Eng-Fre with the modified algorithm. Removing oscillating points (eg. multi-sense word-pairs) now allows other data points (in the same cluster) to move to coherent clusters.

```
flag=1;
while flag do
    INIT Step: Set $q$=2; $BP$=$\phi$; $it$=0;
    INC Step: Compute $U$ with $q$ eigvecs with
    greatest eigvals
        Initialize $q$ centres from rows of $U$
        Initialize $q+1^{th}$ centre as origin
    Elongated $k$-means clustering($U, q$+1):
    if #points assigned to origin $> 0$ then
        if $BP \neq \phi$ then
            Remove rows from U;
            Goto INIT Step;
        else
            q=q++; Goto INC Step;
        end
    else
        $BP = i$; it++;
        if $it > 4$ then
            flag=0;
        end
    end
end
N=BP-4
```

5 Experimental Setup

Since we are interested in improving the performance of low-resource EBMT, the English-Haitian

Lang-Pair	data	Manual	SangAlgo	Mod Algo
Eng-Fre(TM)	10k	0.1777 (10 clusters)	0.1641 (35 clusters)	0.1790 (27 clusters)
Eng-Chi(LM)	30k	0.1290 (110 clusters)	0.1257 (82 clusters)	0.1300 (75 clusters)

Table 1: BLEU scores with templates created using manually, SangAlgo (Sanguinetti et al., 2005) and the modified algorithm to find N.

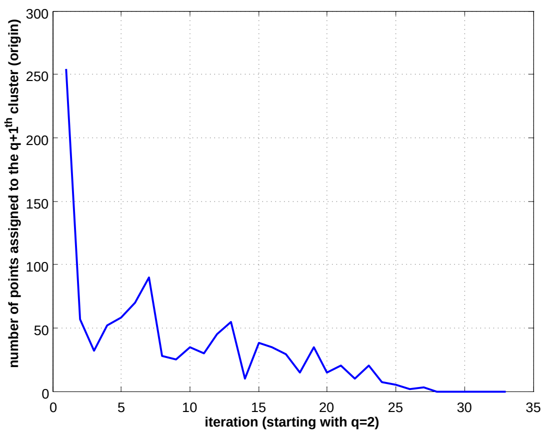

Figure 2: Plot of number of data points assigned to the origin in every iteration using algorithm 4.2 in EBMT for 10k Eng-Fre.

medical domain data (Haitian Creole, CMU, 2010) (1619 sentence-pairs) was used. To test the performance in other languages, we simulated sparsity by choosing less training data from various corpora for 2 more languages pairs, English-Chinese (Eng-Chi) and English-French (Eng-Fre). For the Eng-Chi experiments, we extracted 2 sets of 15k and 30k sentence pairs from the FBIS (NIST, 2003) corpus. The data was segmented using the Stanford segmenter (Tseng et al., 2005). For Eng-Fre, we randomly formed 2 sets of 30k and 100k from the Hansard Corpus (LDC, 1997). Although we are only interested in small data sets, it would be interesting to see the performance of the algorithm with larger data sets as well. For this purpose, we also have a larger set of 200k Eng-Chi. We generalized only words that appeared at least (th_1) times and at most (th_2) times in the training data. Since the idea behind using templates is to obtain longer phrasal matches for phrasal units that contain less frequent words, we chose, th_1=4 and th_2=15. Choosing an even lower th_1 can result in poorly aligned word-pairs. To build the LMs, the target half of the training data was used with Kneser-Ney smoothing. The value of n for the n-gram LMs was chosen as 5. The value of n was not tuned, instead the same value of n was used for building all the LMs. The target words and equivalence class labels of the word-pairs used in building templates in the TM were used to build the template-based LMs.

For tuning the parameters of the EBMT system, 200 sentence pairs were used. For testing, 4000 sentence pairs were used for Eng-Chi and Eng-Fre. Only 200 sentence-pairs were used as test data in the English-Haitian task as the amount of data available was limited. To assess the translation quality, the 4-gram word-based BLEU metric is used. Since BLEU scores have a few limitations, to really know if the templates are helping, we performed a statistical significance test. For this, the 4000 test sentences were split into 10 test files and the Wilcoxon (Wilcoxon, 1945) Signed-Rank test was used to find the statistical significance.

	POS	Auto Clus
TM	0.1283	0.1296
LM	0.1288	0.1300

Table 2: Average BLEU scores with templates from POS and Automatic clusters on 30k Eng-Chi.

6 Results

6.1 Equivalence classes

POS tags are good candidates for equivalence classes. POS tags can be obtained with semi-supervised learning (Tseng et al., 2005) techniques with training data. However, for languages with limited data resources, obtaining POS tags is hard. In such conditions, the question remains, *"Are automatically found clusters as good as POS tags?"*. To answer this, we created templates based on POS tags and compared their performance with templates created using automatically found clusters on 30k Eng-Chi. The POS tags were obtained using Levy et al. (2003) and the templates were applied in the LM and TM. For the POS experiment, the word-pairs for the TM templates were grouped using the POS tags of the target word. For the comparison to be fair, we grouped only those words based on POS tags that were also used in the automatic clustering process. Target words with multiple POS tags were not considered. The BLEU scores with POS templates and templates created using automatic clusters on the 10 test files were almost the same in both the TM and LM case (Table 2). Due to space constraints we only report the average BLEU scores over the test files. Hence, automatically found clusters are good candidates for creating templates in sparse data conditions.

6.2 Incoherent points

Table 3 shows the changes in the cluster members due to removal of incoherent data points with

Impure clusters	Pure clusters
(“almost” *“presque”*)	
(“certain” *“certains”*)	
(“his” *“sa”*)	(“his” *“sa”*)
(“his” *“son”*)	(“his” *“son”*)
(“its” *“sa”*)	(“its” *“sa”*)
(“its” *“ses”*)	(“its” *‘‘ses”*)
(“last” *“hier”*)	
(“my” *“mes”*)	(“my” *“mes”*)
(“my” *“mon”*)	(“my” *“mon”*)
(“our” *“nos”*)	(“our” *“nos”*)
(“our” *“notre”*)	(“our” *“notre”*)
(“their” *“leur”*)	(“their” *“leur”*)
(“their” *“leurs”*)	(“their” *“leurs”*)
(“these” *“ces”*)	(“these” *“ces”*)
(“too” *“trop”*)	
(“without” *“sans”*)	
	(“his” *“ses”*)

Table 3: Cluster purity before and after removal of oscillating points with 10k Eng-Fre (th_1 >9)

10k Eng-Fre. Words that oscillated, (“almost” *“presque”*), (“certain” *“certains”*), (“last” *“hier”*) and (“without” *“sans”*) were removed from the cluster. (“his” *“ses”*) was added to the modified cluster, which is good since other versions of “his” are already present. (“too” *“trop”*), which did not fit well, got placed into a different cluster. The same was observed in Eng-Chi. Word-pairs with wrong alignments, data errors (typos), words with multiple senses that fit in many contexts were found to be removed. For 200k Eng-Chi, 91 word-pairs were discarded and 5 to 11 word-pairs were discarded in all other cases. For Eng-Chi and Eng-Fre, the total number of word-pairs clustered were between 1000 to 3000, and 265 for Eng-Hai.

6.3 Number of clusters (N)

Table 1 compares the average BLEU scores obtained on 10 test files from the empirically found N and automatically found N applied in the TM and LM for 30k Eng-Chi. To find the N empirically, the Spectral Clustering Algorithm in Gangadharaiah et al. (2006) was run with different values of N and the value of N that gave the highest BLEU score on the tune file was chosen. Tuning the parameters for each N took on average 8 days (on a 2.9 GHz dual-core processor). The scores obtained with templates created from automatically found N versus empirically found N is almost the same. Finding the right N is important, “Man. worst” shows the scores obtained with the worst value of N. Table 5 compares the average BLEU scores for Eng-Fre with templates applied in LM with 30k and 100k sentence-pairs.

File		Man. worst	Man. best	Auto
tune	TM	0.1240	0.1335	0.1339
test	TM	0.1248	0.1298	0.1296
tune	LM	0.1280	0.1333	0.1338
test	LM	0.1230	0.1290	0.1300

Table 4: Average BLEU scores on test and tune files with templates created using manually and automatically found N on 30k Eng-Chi.

Data	Baseline	Manual	Auto
30k	0.2100	0.2156	0.2152
100k	0.2210	0.2274	0.2290

Table 5: Comparison of average BLEU scores obtained with templates and baseline with no templates in Eng-Fre.

6.4 Templates in the TM and LM

Table 6 and Table 5 show average BLEU scores obtained by using templates and compares the scores obtained on a baseline system that used no templates. The results clearly show the gains that can be obtained by using templates. The improvements over the baseline were statistically significant. For the 15k and the 30k Eng-Chi cases, templates in the TM are more useful than templates in the LM, whereas, for 200k, templates in the LM are more beneficial. It is known that increasing the amount of training data in an EBMT system with templates in the TM will eventually lead to saturation in performance, where they perform about as well as the system with no templates. This is clearly seen in the results. However, this is not the case with templates in the LM. Template-based LMs continue to give better probability estimates and hence better translation quality even with larger training data sets.

7 Conclusion and Future work

This paper introduced a method for automatically finding the number of clusters (N) for a real world problem. The algorithm also refined the clustering process by removing incoherent points and showed

Lang-Pair		Baseline	LM	TM
Eng-Chi	15k	0.1076	0.1098	0.1102
Eng-Chi	30k	0.1245	0.1300	0.1338
Eng-Chi	200k	0.1905	0.1936	0.1913
Eng-Haitian		0.2182	0.2370	0.2290

Table 6: BLEU scores with templates applied in LM and TM.

that discarding these points boosts the translation quality above the best N found empirically. This paper showed significant improvements by adding generalized templates using the resulting clusters both in the Translation and Language Model over the baseline with no templates.

One of the main goals of the paper was to improve translation quality in EBMT systems working with small data sets. It was seen that templates in the LM continued to show better performance over the baseline even with larger data sets. It would be interesting to see if template-based LMs show the same behavior with even larger training data and we would like to do this as future work.

References

H. U. Block 2000. Example-Based Incremental Synchronous Interpretation. In W. Wahlster (ed.). *Vermobil: Foundations of Speech-to-Speech Translation,* Springer, Heidelberg, pp. 411-417.

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer and P. Roossin. 1990. A Statistical Approach to Machine Translation. *International Conference on Computational Linguistics, 16:79-85.*

R. D. Brown, R. Hutchinson, P. N. Bennett, J. G. Carbonell, P. Jansen. 2003. Reducing Boundary Friction Using Translation-Fragment Overlap. *Ninth Machine Translation Summit,* pp. 24-31.

R. D. Brown. 2000. Automated Generalization of Translation Examples. *International Conference on Computational Linguistics,* pp. 125-131.

I. Cicekli and H. A. Güvenir. 2001. Learning Translation Templates from Bilingual Translation Examples. *Workshop on Example-Based Machine Translation at MT-SUMMT VIII,* pp. 57-76.

R.Gangadharaiah, R.D.Brown and J.Carbonell. 2006. Spectral Clustering for Example Based Machine Translation. *HLT:North American Chapter of the Association for Computational Linguistics,* pp.41-44.

H.Kaji, Y.Kida, Y.Morimoto. 1992. Learning Translation Templates from Bilingual Text. *International Conference on Computational Linguistics,* pp.672-678.

K. Kirchhoff and M. Yang. 2005. Improved language Modeling for Statistical Machine Translation. *Association for Computational Linguistics, Workshop on Building and Using Parallel Texts,* pp. 125-128.

P. Koehn. 2004. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. *The Association for Machine Translation.*

A. Lavie. 2008. Stat-XFER: A General Search-Based Syntax-Driven Framework for Machine Translation. *Conference on Intelligent Text Processing and Computational Linguistics, LNCS 4919,* pp. 362-375.

LDC. 1997. *Hansard Corpus of Parallel English and French.* Linguistic Data Consortium, December. http://www.ldc.upenn.edu/

R. Levy and C. D. Manning. 2003. Is it harder to parse Chinese, or the Chinese Treebank? *Association for Computational Linguistics,* pp. 439-446.

A. Ng, M. Jordan, and Y. Weiss. 2001. On Spectral Clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems,* pp.849-856.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. *Association for Computational Linguistics*. pp. 311-318.

Public release of Haitian Creole language data by Carnegie Mellon, 2010. *http://www.speech.cs.cmu.edu/haitian/*

G. Sanguinetti, J. Laidler and N. D. Lawrence. 2005. Automatic determination of the number of clusters using spectral algorithms. *IEEE Machine Learning for Signal Processing,* pp. 55-60.

H. L. Somers, I. McLean and D. Jones. 1994. Experiments in Multilingual Example-Based Generation. *Conference on the Cognitive Science of Natural Language Processing.*

H. Tseng, P. Chang, G. Andrew, D. Jurafsky and C. Manning. 2005. A Conditional Random Field Word Segmenter. *Fourth SIGHAN Workshop on Chinese Language Processing*.

T. Veale and A. Way. 1997. Gaijin: A Template-Driven Bootstrapping Approach to Example-Based Machine Translation. *New Methods in Natural Language Processing,* pp. 239-244.

D. Verma and M. Meila. 2003. Comparison of Spectral Clustering Algorithms. http://www.ms.washington.edu/ spectral/.

F. Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics, 1,* 80-83.

K. Yamada and K. Knight 2001. A Syntax-based Statistical Translation Model. *Association for Computational Linguistics.* pp. 523-530.

L. Zelnik-Manor and P. Perona. 2004. Self-Tuning Spectral Clustering. *Advances in Neural Information Processing Systems.* pp. 1601-1608.