An Unsupervised Alignment Model for Sequence Labeling: Application to Name Transliteration

Najmeh Mousavi Nejad
Department of Engineering, Islamic Azad University, Science & Research Branch, Punak, Ashrafi Isfahani, Tehran, Iran
najme.mousavi@gmail.com

Shahram Khadivi
Department of Computer Engineering, Amirkabir University of Technology 424 Hafez Ave, Tehran, Iran 15875-4413
khadivi@aut.ac.ir

Abstract

In this paper a new sequence alignment model is proposed for name transliteration systems. In addition, several new features are introduced to enhance the overall accuracy in a name transliteration system. Discriminative methods are used to train the model. Using this model, we achieve improvements on the transliteration accuracy in comparison with the state-of-the-art alignment models. The 1-best name accuracy is also improved using a name selection method from the 10-best list based on the contents of the web. This method leads to a relative improvement of 54% over 1-best transliteration. The experiments are conducted on an English-Persian name transliteration task. Furthermore, we reproduce the past studies results under the same conditions. Experiments conducting on English to Persian transliteration show that new features provide a relative improvement of 5% over previous published results.

1 Introduction

Transliteration is a phonetic translation that finds the phonetic equivalent in target language given a source language word. The quality of name transliteration plays an important role in a variety of applications such as machine translation, as proper nouns are usually not in the dictionary and also new ones are introduced every day (e.g. scientific terms).

The transliteration process consists of training stage and testing stage. In the training stage the model learns segment alignment and produces transformation rules with a probability assigned to each of them. In the test stage it uses these transformation rules to generate the target name. Obviously the alignment process highly affects the results. There are some alignment tools which produce alignments from a bilingual corpus such as GIZA++ (Och and Ney, 2003).

Previous studies can be divided into two categories according to their alignment process: those which apply alignment tools or predefined algorithms in their transliteration process and those that propose new algorithms for aligning word pairs.

There has been an exploration on several alignment methods for letter to phoneme alignment (Jiampojamarn and Kondrak, 2010). M2M-aligner, ALINE which performs phonetic alignment, constraint-based alignment and Integer Programming were investigated. The system was evaluated on several data sets such as Combilex, English Celex, CMUDict, NETTalk, OALD and French Brulex.

Furthermore transliteration based on phonetic scoring has been studied using phonetic features (Yoon et al., 2007). This method was evaluated for four languages – Arabic, Chinese, Hindi and Korean – and one source language – English. The name pairs were aligned using standard string alignment algorithm based on Kruskal.

Substring-based transliteration was investigated applying GIZA++ for aligning name pairs and using open-source CRF++ software package for training the model (Reddy and Waxmonsky, 2009). The model was tested from English to three languages - Hindi, Kannada and Tamil.

English-Japanese transliteration was performed using a maximum entropy model (Goto et al., 2003). First the likelihood of a particular choice of letter chunking into English

Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages 73–81,
Chiang Mai, Thailand, November 12, 2011.

conversion units is calculated and the English word is divided into conversion units that are partial English character strings in an English word. Second each English conversion unit is converted into a partial Japanese character strings called katakana. In this process the English and Japanese contextual information are considered simultaneously to calculate the plausibility of conversion from each English conversion unit to various Japanese conversion candidate units using a single probability model.

There are a few researches which do not use alignment in the transliteration process. For example in recent years two discriminative methods corresponding to local and global modeling approaches were proposed (Zelenko and Aone, 2006). These methods do not require alignment of names in different languages and the features for discriminative training are extracted directly from the names themselves. An experimental evaluation of these methods for name transliteration was performed from three languages (Arabic, Korean, and Russian) into English.

The language pair we perform our tests on, is Persian-English and vice versa. There have been a few researches on Persian language (Karimi et al., 2007). The quality of transliterated names has been improved in the past studies. However, the proposed method is language specific and the algorithm is designed for Persian language. The best general language independent model in the mentioned paper is CV-MODEL3. To compare our new method, we have reproduced its results under similar conditions. In both systems the same corpus was used and both experiments are 10-fold cross-validation.

In this paper, the openNlP maximum entropy package is used for training the model[1]. We define new features for discriminative training. Moreover a new approach for aligning name pairs is proposed. In the case studies, we investigate the effect of each feature by adding it to and removing it from training process. As a result, the best combination of features is achieved for English-Persian language pair. In addition, we compare our proposed alignment method to GIZA++. Our main concern is finding an alignment model for transliteration. We have found that the most common word alignment tool for transliteration alignment is GIZA++ (Hong, et al., 2009; Karimi, et al., 2007; Sravana Reddy and Sonjia Waxmonsky, 2009). The proposed language-independent alignment method performs similar to GIZA++ results in Top-1 for English-Persian transliteration and improves the accuracy and MRR[2] in Top-5 and Top-10. For reverse transliteration (Persian to English), new alignment shows a significant improvement over GIZA++ outcome. Furthermore an approach based on name frequencies in the web contents is applied to choose one name from 10 best possible transliterations. Since the dominant language of web is English, the experiments were performed for Persian-to-English transliteration and not English-to-Persian.

The rest of this paper is organized as follows: The feature set is described in Sec. 2. The proposed alignment method is described in Sec. 3. In Sec. 4 our experimental study is described. Choosing one name from 10 best transliterations is described in Sec. 5 and the conclusion is described in Sec. 6.

2 Feature Set

Maximum entropy models use features for maximizing log likelihood. Consequently defining proper features has a high impact on the final results. We define two types of features which are binary-valued vectors. For both types of features (consonant-vowel and n-gram), current context (current letter), two past and two future contexts (neighboring letters) are used. We choose a window with a length of 5, since experiments show that lower length or higher length would have degrade the results.

2.1 Consonant-Vowel Features

Every language has a set of consonant and vowel letters. The consonant letters can be divided into different groups based on their types (Table 1).

Plosive (stop)	p , b , t , d , k , g , q
Fricative	f , v , s , z , x , h
Plosive-Fricative	j , c
Flap (tap)	r
Nasal	m , n
Lateral approximant	l , y

Table 1. Six group of consonants

Most combinations of consonant-vowel features were tested for English-Persian language pair. We have found the following consonant-vowel features are the most effective ones for

[1] Available at http://incubator.apache.org/opennlp/

[2] Mean Reciprocal Rank

generating current target letter (t_n). S_i is used to represent the source name characters and t_i represents the target name characters. CV is an abbreviation for consonant- vowel.

$f1_{cv}$: CV_{s_n}
$f2_{cv}$: $CV_{s_{n-1}}$ CV_{s_n}
$f3_{cv}$: $CV_{t_{n-1}}$ $CV_{s_{n-1}}$ CV_{s_n}
$f4_{cv}$: $CV_{s_{n-1}}$ CV_{s_n} $CV_{s_{n+1}}$
$f5_{cv}$: $CV_{s_{n-2}}$ $CV_{s_{n-1}}$ CV_{s_n}
$f6_{cv}$: CV_{s_n} $CV_{s_{n+1}}$ $CV_{s_{n+2}}$
$f7_{cv}$: $CV_{t_{n-1}}$ $CV_{s_{n-1}}$ CV_{s_n} $CV_{s_{n+1}}$
$f8_{cv}$: $CV_{t_{n-1}}$ $CV_{s_{n-2}}$ $CV_{s_{n-1}}$ CV_{s_n}

We have defined three types of CV features. CV-TYPE1 is some basic features to reproduce past studies results. These features consist of $f1_{cv}$, $f2_{cv}$, $f5_{cv}$ and $f6_{cv}$. To achieve better results, some new features are presented called CV-TYPE2 which is an augmented set of features including $f1_{cv}$ to $f8_{cv}$. Finally to track the effect of new consonant grouping strategy, CV-TYPE3 is defined which is similar to CV-TYPE2 except that the consonant letters are divided according to Table 1.

Table 1 can be used for categorizing any language letters as well, by replacing each English letter with its corresponding letter in the target language. These features improve transliteration, but still are not sufficient. Therefore we need n-gram features.

2.2 N-gram Features

In n-gram features for source name, two past and two future contexts are used (a window with a length of 5). For target name however, only two past contexts are used (because we don't have future context yet). Since the maximum entropy is used for training, the whole approach for target name can be considered as Maximum Entropy Markov Model (MEMM) which is a simple extension of the ME classification and is useful for modeling sequences as it takes into account the previous classification decision. But for source name the future letters are known and are used for feature extraction. So the MEMM concept cannot be broadcast to source name as well.

Using S to demonstrate the source name and T to demonstrate the target name, the n-gram features for each name can be summarized as:

s_{n-2} s_{n-1} s_n s_{n+1} s_{n+2}
t_{n-2} t_{n-1} × × ×

For any language pair, all combinations of s_i and t_i can be used to define a feature. In our model, the following set of features has been used:

f1: s_n
f2: s_{n-1} s_n
f3: s_n s_{n+1}
f4: s_{n-2} s_{n-1} s_n
f5: s_{n-1} s_n s_{n+1}
f6: s_n s_{n+1} s_{n+2}
f7: t_{n-1}
f8: t_{n-2} t_{n-1}

The best sequence of above features, varies from one language pair to another. We report the best combination for English-Persian language pair in Sec. 4.

3 The Proposed Alignment Method

Features explained in the previous section, are extracted from the aligned names. In other words, first the alignments of source and target names should be produced. Our proposed alignment method is a two-dimensional Cartesian coordinate system. The horizontal axis is labeled with the source name and the vertical axis is labeled with the target name (or vice versa). A line is drawn from the coordinate (0,0) to the point with coordinate (source_name_length , target_name_length). We mark the corresponding cell in each column of the alignment matrix which has the less distance to the line (Figure 1). Considering Figure 1 the following alignments are achieved:

(a,ا) , (b,ب) , (r,ر) , (a,ا) , (m,م) , (s,ز)

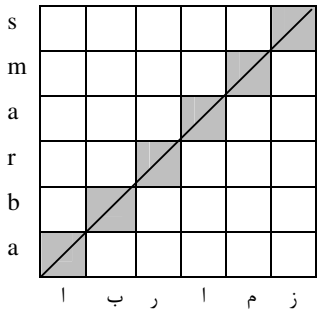

Figure 1. Alignment matrix of (abrams,ابرامز)

The name pair in Figure 1 has a simple alignment. For more complex alignments, some fixed points are needed in order to draw the lines. These fixed points are coordinates of segments that are known to be always alignments of each other. For instance in English-Persian, "ب" is always aligned to "b" or "bb". If there exists any fixed point in the name pair, one line is drawn

from origin to the fixed point coordinate and the other one is drawn from the fixed point to the point with (source_name_length , target_name_length) coordinate. In other words if there are n fixed points in the name pair, there will be n+1 lines in the plane. In Figure 2, (bb,ب) and (n,ن) are fixed points. So the following alignments are achieved:

(g,گ) , (i,ی) , (bb,ب) , (o,و) , (n,ن) , (i,ی)

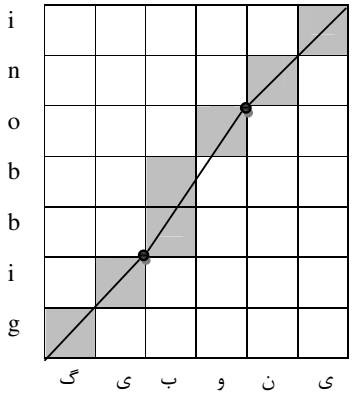

Figure 2. Alignment matrix of (gibboni,گیبونی)

These fixed points help us to perform the alignment process more accurately. The more accurate they are, the better the final results are.

Finding fixed points is difficult for some language pairs, especially for the ones about which we have no knowledge. Based on the fact that our goal is to design a language independent transliteration system, an automatic way to find the fixed points is of interest.

We investigate two approaches for finding the fixed points. In the first one, Moses, a statistical machine translation system is used to define the fixed points. Moses trains translation models for any language pair automatically (Koehn, et al., 2007). In translation process, it produces a phrase table which contains source and target phrases with different lengths and the conditional probability of those phrases. If each letter in transliteration is considered as a word and each name as a sentence, Moses can be used to find the fixed points automatically.

To produce the phrase table, Moses should be run on a bilingual corpus. Any corpus containing name pairs can be used. Then the phrase table is parsed and the phrases with the maximum probabilities are extracted. The length of the phrases is usually between 1 and 3, since for most natural languages, the maximum length of corresponding phonemes of each grapheme is a digraph (two letters) or at most a trigraph (three letters). Once a set of fixed points are found for a language pair, they stay constant for all other transliterations and do not change. In other words it is sufficient to run Moses one time and use produced fixed points for any transliteration task related to that language pair.

In the second approach the training dataset helps the system find the fixed points set. We introduce FPA algorithm which is an unsupervised approach that adopts the concept of EM training. In the expectation step the training name pairs are aligned using current model and in the maximization step the most probable alignments are added to the fixed points set. The algorithm is as follows:

1. An initial and inaccurate alignment is considered, assuming just one line in the alignment matrix.
2. The discriminative model learns the mapping between source and target names using maximum entropy.
3. Using the trained model (ME) and extracting the most probable mappings, an initial set of fixed points are nominated. This process is repeated until the algorithm converges.

A brief sketch of FPA algorithm is presented in Figure 3. In line 2 we initialize the fixed points with an empty set. Line 3 shows the convergence of the algorithm. It means when the fixed points set do not change, the final set is found. In line 6 name pairs with equal lengths are only considered. The corresponding consonant-vowel sequences of the name pairs are generated. If the CV sequences are exactly similar to each other, the name pair is included in the training stage. Although the whole training data can be used in the first iteration, this condition produces a reasonable result with the advantage of ignoring a large amount of training data and saving the time in the first iteration. Line 11 to 21 shows the process of updating the fixed points set. In line 14 forcedAlignment means using current ME model to transliterate source name with the condition in which the produced transliterations should be the same as the target name. This condition guarantees the convergence of the algorithm. Suppose the source name length is J and the target name length is I, then the decoding process is as follows:

1. For each letter of the source name choose top N transformation rules with highest probabilities which lead to producing the

target name.

2. Build a search tree: add N 3-tuple (current letter, generated transliteration, transformation probability) to an N-complete tree.
3. Do beam search to find the best path in the tree. (Best path is the highest multiplication of edges probability).
4. Update set A:

$$FP(S_1^J, T_1^I, A) = \{ (S_{j1}^{j2}, T_i) : \forall j_1 \le j \le j_2 : (j, i) \in A \}$$

$$FP(S_1^J, T_1^I, A) = \{ (S_j, T_{i1}^{i2}) : \forall i_1 \le i \le i_2 : (j, i) \in A \}$$

We change the value of N between 1 and 5. Results show that there is no significant improvement after N = 3 (N > 3). Also time complexity and memory usage increases exponentially. Therefore the best value for N is 3. Line 17 and 18 are final steps in producing fixed points set. |k| is the number of distinct segments in the best path set and $p(\tilde{T}_k|\tilde{S}_k)$ is the probability of $\tilde{T}_k|\tilde{S}_k$ transformation rule. Once the probabilities are calculated, they are compared to a predefined threshold. If they are bigger than threshold, they are added to the fixed points set. We change the value of threshold between 0.7 and 1, and find out the best value for threshold is 0.9. The test stage starts after finding the final fixed points set. The decoding process in test stage is similar to forcedAlignment, but here the condition for generated transliteration (forcing algorithm to produce target name) is meaningless. So any transliteration can be added to Top-N results.

A good method for finding the fixed points generates a set similar to other methods. For example both approaches introduced in this section, lead to similar results. That's why we present only the second approach results in the experiment section. From another point of view, it is sufficient to find the fixed points set for each language pair only once. Because the fixed points set which is found by a proper corpus, is very similar to the set produced by a different corpus on the same language pair. Therefore if more than one set are produced using different corpora, the intersection of these sets is considered as the final fixed points set for other transliteration tasks regarding that language pair.

```
 1: Algorithm FPA
 2: fixedPoints = { } , oldFixedPoints = { }
 3: while( fixedPoints != oldFixedPoints) {
 4:    oldFixedPoints = fixedPoints;
 5:    if( first iteration){
 6:       fixedPoints = updateFixedPoints(names_with_equal_CV_sequence)
 7:    }else{
 8:       fixedPoints = updateFixedPoints(whole_training_corpus)
 9:    }
10: }
11: Function updateFixedPoints(training_data){
12:    bestPathEdges = { };
13:    for( all name pairs) do {
14:       A = forcedAlignment(sourceName, targetName, currentModel)
15:    }
16:    for (all segment pairs in A) do{

17:       p(T̃_k|S̃_k) = Σ p(T̃_k, S̃_k) / Σ_T́ p(T́, S̃_k)
18:       p(S̃_k|T̃_k) = Σ p(T̃_k, S̃_k) / Σ_Ś p(Ś, T̃_k)
19:    }
20:    if (p > threshold) { add transformation rule to the fixedPoints }
21: }
```

Figure 3. Sketch of FPA algorithm

We present a list of most common fixed points for English to Persian transliteration which is sorted in descending order of the probability values.

{ (م mm) , (د dd) , (ب bb) , (و wh) , (ر rr) , (کس x) , (ن kn) , (ن nn) , (ف ff) , (ت tt) , (پ pp) , (ل ll) , (ه h) , (ن n) , (ر r) , (د d) , (گ g) , (ب b) , (ت t) , (ش sh) , (پ p) , (ل l) , (م m) , (ج j) , (ف ph) , (س ss) , (ز z) , (و w) , (ک q) , (ف f) , (و v) , (ی y) , (س s) , (ک k) }

There is a study on statistical machine translation which combines discriminative training and Expectation-Maximization (Fraser and Marcu, 2006). The proposed EMD algorithm uses discriminative training to control the contributions of sub-models. Furthermore, EM is applied to estimate the parameters of sub-models. In contrast to their method, we generate fixed points set by Expectation-Maximization and no parameter estimation is done during EM. The new fixed points set, updated in EM step, improves the alignment quality and consequently causes the model to reestimates its parameters.

4 Case Studies

Two types of experiments have been performed, one for effectiveness of different features and the other for the effectiveness of alignment process. A corpus consisting of 16760 word pairs has been used. These words are names of geographical places, people and companies. This is the same corpus which previous study experiments were performed on (Karimi et al., 2007). Each name has only one transliteration. Many words of different language origins (such as Arabic, French, and Dutch) were included in the corpus. This corpus is referred to as B^+. The experiments apply 10-fold cross-validation in which the whole corpus is partitioned into 10 disjoint segments. This type of experiment is an alternative method for controlling over-fitting.

4.1 Effectiveness of Features

All combinations of f1 to f8 for English-Persian language pair were tested. Table 2 shows mean word accuracy in 10-fold, for English-Persian transliteration. The first row in Table 2 shows reproducing CV-MODEL3 results using some basic features. Extending CV-TYPE1 features to CV-TYPE2 improves the accuracy (second row). Similarly applying the new grouping of consonant letters (CV-TYPE3), leads to a relative improvement of 1% over CV-TYPE2 (third row). CV-TYPE1, CV-TYPE2 and CV-TYPE3 are explained in Sec. 2.2.

The best word accuracy in Table 3 is 58.4%. Comparing word accuracies, it can be concluded that for English-Persian transliteration, the following features are the most effective ones:

f1: s_n
f2: $s_{n-1}\ s_n$
f3: $s_n\ s_{n+1}$
f5: $s_n\ s_{n+1}\ s_{n+2}$
f7: t_{n-1}

As we can see, t_{n-2} does not help in better transliteration. Because written Persian omits short vowels, and only long vowels appear in texts. So t_{n-2} is completely irrelevant for generating current Persian letter. Using f2 and f3 simultaneously, improves the results much more than f4, f5 or f6 alone. Since each of them has the power of bigram feature and together, they provide trigram features.

Experimental results for English to Persian transliteration show that CV-TYPE3 has the best word accuracy among all other consonant-vowel grouping strategy. Therefore, we use this type of consonant-vowel features for the reverse direction as well. Furthermore English-Persian experiments imply n-gram features with a distance of two letters are not useful for Persian names. This is due to Persian language nature. This fact reduces the number of experiments, since it removes f4, f5 and f6 from n-gram features. Table 3 shows the effect of several feature combinations on mean word accuracy in Top-1 for Persian-English transliteration task. The best word accuracy in Table 3 is 20.6%. Therefore, the following features result in best performance.

f1: s_n
f2: $s_{n-1}\ s_n$
f3: $s_n\ s_{n+1}$
f7: t_{n-1}
f8: $t_{n-2}\ t_{n-1}$

Persian-English transliteration is more difficult than English-Persian. Because moving from the language with smaller alphabet size to the one with larger size, increases the ambiguity. Using web page contents improves the transliteration. The strategy is explained in Sec. 5.

f1	f2	f3	f4	f5	f6	f7	f8	CV-TYPE1	CV-TYPE2	CV-TYPE3	Mean WA
√	√	√	×	×	√	×	×	√	×	×	55.3
√	√	√	×	×	√	×	×	×	√	×	56.7
√	√	√	×	×	√	×	×	×	×	√	57.2
√	×	×	×	√	×	√	×	×	×	√	57.3
√	√	√	×	×	×	×	×	×	×	√	57.3
√	√	√	√	√	√	√	√	×	×	√	57.3
√	√	√	√	×	×	√	×	×	×	√	57.4
√	×	×	×	√	√	√	×	×	×	√	57.5
√	√	√	×	×	×	√	√	×	×	√	58.0
√	√	√	×	√	√	√	×	×	×	√	58.2
√	√	√	×	×	×	√	×	×	×	√	58.4
√	√	√	×	×	√	√	×	×	×	√	58.4

Table 2. The effect of several feature combinations on mean word accuracy in Top-1 for English-Persian transliteration

f1	f2	f3	f7	f8	CV-TYPE3	WA
√	√	√	×	×	×	17.1
√	√	√	√	×	√	19.3
√	√	×	√	√	√	19.8
√	×	√	√	√	√	20.5
√	√	√	√	√	√	20.6

Table 3. The effect of several feature combinations on mean word accuracy in Top-1 for Persian-English transliteration

4.2 Effectiveness Of Alignment

The proposed alignment (FixedPointsAlign) results are compared to GIZA++ alignment. The settings of important parameters of GIZA++ are as follows: five iterations for each IBM1 model and HMM and three iterations for each IBM3 and IBM4 models. We checked GIZA++ output for name pairs and discovered the alignments are always monotone, except for rare cases. That's why it is used in past studies as well (Hong, et al., 2009; Karimi, et al., 2007; Sravana Reddy and Sonjia Waxmonsky, 2009). The approaches using GIZA++ utilize symmetrized alignments in both directions. All of the experiments are done on B^+ corpus, using 10-fold cross validation. The results are compared to CV-MODEL3 (Karimi et al., 2007). The most effective features, founded in the previous section, are included in the training stage. These combinations are specifically appropriate for English-Persian language pair. For other languages if the best combination is not known, all the features, f1 to f8 should be included in the feature extraction.

For each fold, word accuracy and MRR is computed. Table 4 and Table 5 show mean word accuracy and mean MRR in Top-1, Top-5 and Top-10 for English to Persian. Persian to English results are presented in Table 6 and Table 7.

The transliteration systems that use GIZA++ in their alignment differ from each other by transliteration generation process. Since GIZA++ has a unique strategy for aligning sentences or name pairs. CV-MODEL3 is a language-independent model which uses GIZA++ for aligning name pairs. Since our experiment

conditions are exactly the same as CV-MODEL3 experiments conditions, the results are comparable.

Table 4 shows that our proposed alignment method is a proper replacement for GIZA++ tool. It has an equal accuracy in Top-1 and also improves accuracy in Top-5 and Top-10 transliterations. SLA (single line align) is the proposed method with an empty fixed points set. As can be seen from Table 4, defining a proper set of fixed points significantly improves the results. Furthermore Table 6 and Table 7 show that for Persian to English transliteration, our proposed alignment algorithm significantly improves the results. The outcomes lead us to the conclusion that although GIZA++ provides good results in English to Persian transliteration, it does not produce a reasonable result in the reverse direction. This is due to parameters setting. Unlike our proposed alignment, GIZA++ alignment is highly dependent to its parameters.

N-Best	SLA	CV-MODEL3	GIZA++	FPA
Top-1	50.7	55.3	58.4	58.4
Top-5	77.0	84.5	86.8	88.7
Top-10	84.0	89.5	90.8	92.6

Table 4. Mean word accuracy of 10-fold on B^+ corpus for English to Persian transliteration

N-Best	GIZA++	FPA
Top-1	58.4	58.4
Top-5	70.2	70.9
Top-10	70.7	71.5

Table 5. Mean MRR of 10-fold on B^+ Corpus for English to Persian.

N-Best	SLA	CV-MODEL3	GIZA++	FPA
Top-1	19.4	17.6	14.6	20.6
Top-5	41.6	36.2	32.7	44.9
Top-10	50.4	46.0	38.4	53.2

Table 6. Mean word accuracy of 10-fold on B+ corpus for Persian to English transliteration

N-Best	GIZA++	FPA
Top-1	14.6	20.6
Top-5	21.3	29.7
Top-10	22.1	30.8

Table 7. Mean MRR of 10-fold on B+ Corpus for Persian to English

5 N-Best Reranking

Generating 10 best transliterations instead of one name definitely has a better word accuracy, because if the target name exist in one of the 10 names, the word accuracy is equal to one for that name pair. But an efficient transliteration system should produce only the correct ones.

A large corpus containing several names can be considered as a reference to choose one name from possible transliterations. First the unigram probability of each transliteration is calculated. Then the transliteration with the max probability is chosen as the final result. Since the dominant language in the web is English, it is the best corpus for Persian to English transliteration. As a result, in this section the experiments were performed for Persian-to-English transliteration and not English-to-Persian.

We calculate the probabilities of Top-10 for each source name and the one with the maximum probability is chosen as the final transliteration. A test file consisting of 1676 name pairs was produced. We extract 10% of the train file randomly to generate the test file. The word accuracy of this approach is **32.1%** and the accuracy for the same test and train files, generating only one transliteration (Top-1) is **20.8%**. It means that this approach leads to a relative improvement of **54%** over Top-1 results.

6 Conclusions

In this paper, we presented a language-independent alignment method for transliteration. Discriminative training is used in our system. The proposed method has improved transliteration generation compared to GIZA++. Furthermore we defined a number of new features in the training stage.

For Persian to English transliteration, web pages contents are used to choose one name from 10-best hypothesis list. This approach leads to a relative improvement of 54% over simple Top-1 transliteration.

References

Fraser, A., Marcu, D., Semi-Supervised Training for Statistical Word Alignment, Proceedings of ACL-2006, pp. 769-776, Sydney, Australia

Goto, I., Kato, N., Uratani, N., Ehara, T., Transliteration Considering Context Information Based on the Maximum Entropy Method, In Proc. Of IXth MT Summit. (2003)

Hong, G., Kim, M., Lee, D., Rim, H., A Hybrid to English-Korean Name Transliteration, Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 92–95, Suntec, Singapore, 7 August 2009.

Jiampojamarm, S., Kondrak, G., Letter-Phoneme Alignment: An Exploration, Proceedings of the 48th Annual Meet-ing of the Association for Computational Li-nguistics, pages 780–788, Uppsala, Sweden, 11-16 July 2010.

Josef, F., Ney. H., Discriminative Training and Maximum Entropy Models for Statistical Machine Translation, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 295-302.

Josef, F., Ney. H., A Systematic Comparison of Various Statistical Alignment Models, Computational Linguistics, vol.29 (2003), pp. 19-51

Karimi, S., Scholer, F., Turpin, A., Collapsed Consonant and Vowel Models: New Approaches for English-Persian Transliteration and Back-Transliteration, The 45th Annual Meeting of the Association for Computational Linguistics (ACL'07), pages 648-655, Prague, Czech Republic, June 2007.

Karimi, S., Machine Transliteration of Proper Names between English and Persian, A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy, BEng. (Hons.), MSc.

Koehn, P., Hoang, H., Birch A., CallisonBurch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer C., Bojar, O., Constantin, A., Herbst E., 2007. Moses: Open source toolkit for statistical machine translation. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07) – Companion Volume, June.

Microsoft Web N-gram Services available at http://research.microsoft.com/en-us/collaboration/focus/cs/web-ngram.aspx

Reddy, S., Waxmonsky, S., Substring-based Transliteration with Conditional Random Fields, Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 92–95, Suntec, Singapore, 7 August 2009.

Yoon, S., Kim, K., Sproat, R., "Multilingual Transliteration Using Feature based Phonetic Method", Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 112–119, Prague, Czech Republic, June 2007, Association for Computational Linguistics.

Zelenko, D., Aone. C., Discriminative Methods for Transliteration, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 612–617, Sydney, July 2006.