Semi-supervised Transliteration Mining from Parallel and Comparable Corpora

Walid Aransa, Holger Schwenk, Loic Barrault

LIUM, University of Le Mans
Le Mans, France
firstname.lastname@lium.univ-lemans.fr

Abstract

Transliteration is the process of writing a word (mainly proper noun) from one language in the alphabet of another language. This process requires mapping the pronunciation of the word from the source language to the closest possible pronunciation in the target language. In this paper we introduce a new semi-supervised transliteration mining method for parallel and comparable corpora. The method is mainly based on a new suggested Three Levels of Similarity (TLS) scores to extract the transliteration pairs. The first level calculates the similarity of of all vowel letters and consonants letters. The second level calculates the similarity of long vowels and vowel letters at beginning and end position of the words and consonants letters. The third level calculates the similarity consonants letters only.

We applied our method on Arabic-English parallel and comparable corpora. We evaluated the extracted transliteration pairs using a statistical based transliteration system. This system is built using letters instead or words as tokens. The transliteration system achieves an accuracy of 0.50 and a mean F-score 0.8958 when trained on transliteration pairs extracted from a parallel corpus. The accuracy is 0.30 and the mean F-score 0.84 when we used instead a comparable corpus to automatically extract the transliteration pairs. This shows that the proposed semi-supervised transliteration mining algorithm is effective and can be applied to other language pairs. We also evaluated two segmentation techniques and reported the impact on the transliteration performance.

1. Introduction

Transliteration is the process of writing a word (mainly proper noun) from one language in the alphabet of another language. This process requires mapping the pronunciation of the word from the original language to the closest possible pronunciation in the target language. Both the word and its transliteration are called a Transliteration Pair (TP). The automatic extraction of TPs from parallel or comparable corpora is called Transliteration Mining (TM). The transliteration pairs are important for many applications like Machine Translations (MT), machine transliteration, cross language information retrieval (IR) and Name Entity Recognition (NER). For example, in MT, TM can be used to improve the word alignments, or to train a system to transliterate proper nouns in out-of-vocabulary (OOV) words. In machine transliteration, the obtained TPs are used to train statistical transliteration system, while in IR, it is used to enrich the search results with orthographical variations.

Recently, TM has gained considerable attention from the research community. There are several methods to perform TM: supervised, unsupervised and semi-supervised. Also, some TM researches focus on parallel corpora and others on comparable corpora. In this paper we will focus on semi-supervised method with both parallel corpora and comparable corpora.

We applied our method on an Arabic-English transliteration task using letter based SMT system trained on the extracted transliteration pairs. Then, we used this transliteration system in our semi-supervised method to extract transliteration pairs from comparable corpora. Although this work focuses on Arabic-English, it can be applied to any language pair. We are conducting this research in the context of MT, in order to decrease the OOV rate in the translation task.

There are several challenges related to Arabic transliteration. One of the challenges is that some Arabic letters have no phonically equivalent letters in English (e.g. ض and ط), and also some English letters do not have phonically equivalent letters in Arabic (e.g. v). Another challenge is the missing of short vowels (i.e. diacritics) in the Arabic text, while it should be mapped to existing letters in English text during the transliteration process. Additionally, some Arabic letters can be mapped to any letter from a group of phonically close English letters (e.g. ب to p or b), and some Arabic letters can be mapped to a sequence of English letters (e.g. خ to 'kh'). There is also a tokenization challenge, since unlike English, sometimes, the Arabic name is concatenated to one clitic (e.g. preposition ب or conjunction و) or both together (e.g. وب), which requires an advanced detection and seg-

mentation for these clitics before performing the transliteration.

There are two types of transliteration, forward and backward. In forward transliteration, the names are transliterated from its original language to another language, like the Arabic origin name "محمد" transliterated to "Mohamed" in English. In backward transliteration, the transliterated names are transliterated back to the origin names in its original language, like "بوش" will be transliterated back to "Bush". For simplicity, in this paper we will not differentiate between forward transliteration and backward transliteration. In future work, we will focus on addressing the specific problems related to each transliteration type.

The paper is organized as follows: the next section presents related work, followed by a description of the TM algorithm when using parallel corpora. This technique is extend to comparable corpora in section 4. The paper concludes with a discussion of the perspectives of this work.

2. Related work

The related work includes TM and transliteration research. For TM, there are several methods to perform it, supervised, unsupervised and semi-supervised. Also, some TM researches focus on parallel corpora and others on comparable corpora. [1] uses variant of the SOUNDEX methods and n-grams to improve precision and recall of name matching in the context of transliterated Arabic name search. Original, SOUNDEX was developed by [2] which is an algorithm used for indexing names by sound as pronounced in English. The SOUNDEX code for a name consists of a letter followed by three numerical digits: the letter is the first letter of the name, and the digits encode the remaining consonants. Similar sounding consonants share the same digit. For example, the labial consonants B, F, P, and V are each encoded as the number 1. The method proposed by [1] reduces the orthographical variations by 30% using SOUNDEX improved precision slightly but they observed a decrease in recall. [3] presents two methods for improving TM, phonetic conflation of letters and iterative training of a transliteration model. The first method is an improved SOUNDEX phonetic algorithm. They propose SOUNDEX like conflation scheme to improve the recall and F-measure. Also iterative training method was presented that improves the recall but decreases the precision.

[4] presents an adaptive learning framework for Phonetic Similarity Modeling (PSM) that supports the automatic construction of transliteration lexicons. PSM measures the phonetic similarity between source and target words pairs. In a bi-text snippet, when an source language word EW is spotted, the method searches for the word's possible target transliteration CW in its neighborhood. EW can be a single word or a phrase of multiple source language words. In this paper, they initialize the learning algorithm with minimum machine transliteration knowledge, then it starts acquiring more transliteration knowledge iteratively from the Web. They study the active learning and the unsupervised learning strategies that minimize human supervision in terms of data labeling. They report that the unsupervised learning is an effective way for rapid PSM adaptation while active learning is the most effective in achieving high performance. Another TM method relies on a Bayesian technique proposed by [5]. This method simultaneously co-segments and force-aligns the bilingual segments through rewards the re-use of features already in the model. The main assumption that transliteration pairs can be derived by using bilingual sequence pairs already learned by the model, or by introducing a very short unobserved pair into the derivation. They assume that incorrect pairs are likely to have large contiguous segments that are costly to force-align with the model. The transliteration classifier is trained on features derived from the alignment of the candidate pair as well as other heuristic features. They report a results indicate that transliteration mining of English-Japanese using this method should be possible at high levels of precision and recall. [6] adapts graph reinforcement to work with large training sets. They introducs parametrized exponential penalty to formulation of graph reinforcement which led to improvement in precision. They report that TM quality using comparable corpora is impacted by the presence of phonically similar words in comparable text, so they extracted the related segments that have high translation overlap and used them for TM, which leads to higher precision for the suggested TM methods. An automatic language pair independent method for transliteration mining using parallel corpora is proposed by [7]. They models transliteration mining as interpolation of transliteration and non-transliteration sub-models. Two methods, unsupervised and semi-supervised were presented with the results that show that semi-supervised method is out performing unsupervised method.

For transliteration research, [8] uses two algorithms based on sound and spelling mappings using finite state machines to perform the transliteration of Arabic names. They report that transliteration model can be trained on relatively small list of names which is easier to obtain than training data needed for training phonetic based models. [9] presents DirecTL, a language independent approach to transliteration. DirecTL is based on an online discriminative sequence prediction model that employes EM-based many-to-many unsupervised alignment between target and source. While, [10] uses a joint source channel models on the automatically aligned orthographic transliteration units of the auto-

matically extracted TPs. They compare the results with three online transliteration systems and reported better results.

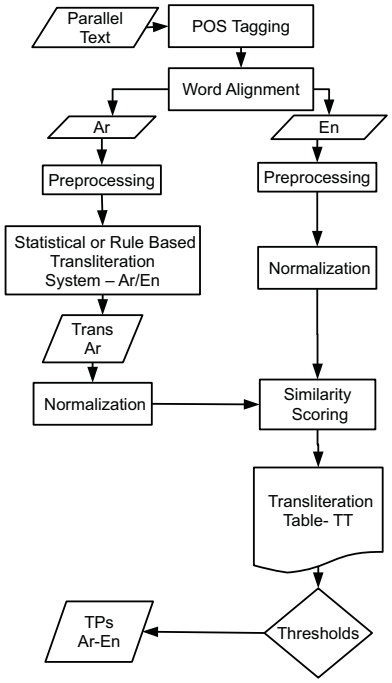

Figure 1: Extracting TPs from parallel corpora

3. Transliteration mining using parallel corpora - semi-supervised

In this section, we will introduce a corpus based computational method to extract TPs from parallel corpus. In order to evaluate the extracted pairs, we trained a letter based statistical transliteration system on TPs and evaluate the system performance which is correlated with the transliteration mining quality.

3.1. TM algorithm for parallel corpora

The algorithm as shown in Figure 1 is designed to compare two aligned words and detect the words which are transliteration of each other, with respect to the observations in section 3.3. We developed the following TM algorithm:

(1) First, the parallel corpus is tagged using a part-of-speech (POS) tagger. We used Stanford POS tagger [11] for English and Mada/Tokan [12] for Arabic POS tagging.

(2) Then, we align the tagged bitext using Giza++ [13], using the source/target alignment file, remove all aligned word pairs with POS tags other than noun (NN) or proper noun (PNN) tags and remove all English words starting with lower-case letters. Words which have most lowest alignment scores are removed (about 5% from the total number of aligned word pairs).

(3) After that removing the POS tags from Arabic and English words.

(4) Then, transliterate the Arabic word A into English using a rule based transliteration system (or a previously trained statistical based transliteration system).

(5) Normalize the transliteration of Arabic word A_t as well as the English word to $Norm_1$, $Norm_2$ and $Norm_3$ as explained in section 3.2. The objective of the normalization is folding English letters with similar phonetic to the same letter or symbol.

(6) For each aligned Arabic transliterated word A_t and English word E, use their normalized forms to calculate the three levels of similarity scores which we store in a transliteration table (TT).

(7) Extract TPs from the TT by applying a threshold on the three levels similarity scores. We selected the thresholds using empirical method shown in section 3.5.4.

3.2. English normalization and three levels similarity scores for TM

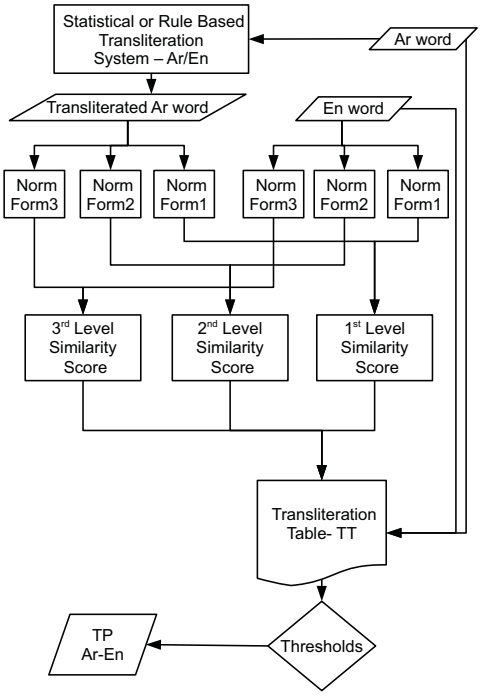

Figure 2: Calculating the three levels of similarity scores

As shown in Figure 2, we developed a three normalization functions which can be used to normalize the Arabic transliterated word and English word to be more comparable to each other phonically. These normalized forms are used to

calculate the similarity between the transliterated word and the English word based on three levels of similarity. The first level calculates the similarity of all vowel letters and consonants letters. The second level calculates the similarity of long vowels and vowel letters at beginning and end position of the words as well as consonants letters. The third level calculates the similarity of consonants letters only. The details of each normalization function as following:

(1) $Norm_1$ normalization function: Normalize the transliteration of Arabic word as well as the English word. The objective of the normalization is folding English letters with similar phonetic to one letter or symbol. In $Norm_1$, all letters are converted to lower case, phonically equivalent consonants and vowels are folded to one letter (e.g. p and b are normalized to b, v and f are normalized to f, i and e are normalized to e), double consonants are replaced by one letter, and finally a hyphen "-" is inserted after the initial two letters "al" -which is the transliteration of the usually concatenated Arabic article "ال"- if it is not already followed by it.

(2) $Norm_2$ normalization function: Using $Norm_1$ output, double vowels are replaced by one similar upper-case letter (i.e. ee is normalized to E), remove non-initial and non-final vowels only if not followed by vowel or not preceded by vowel.

(3) $Norm_3$ normalization function: Using $Norm_2$, hyphen - and vowels are removed.

Hence, for each Arabic word A and English word E. if A_t is the transliteration of A into English, we can calculate the following three levels similarity scores while i=1,2,3

$$TLS_i = \frac{Levenshtein(Norm_i(A_t), Norm_i(E))}{|Norm_i(E)|} \quad (1)$$

In this formula, Levenshtein function is the edit distance between the two words, which is the number of single-character edits required to change the first word into the second one.

3.3. Customized English pronunciation similarity comparison for Arabic-English transliteration

Our TM algorithm is based on the following pronunciation (and hence transliteration) observations in the English language considering the transliteration task from Arabic language characteristics:

1. In most cases, we can sort the letter's impact on transliteration from low to high as following:
   - Phonically similar vowels have low impact.
   - Phonically dissimilar vowels have medium impact.
   - Consonants letters have significant impact.
2. The double vowels produce long vowel sound have more impact on the pronunciation of the English word.
3. The sequence of two or more different vowel letters, has a special pronunciation which has more impact on the pronunciation of the English word.
4. The vowel at the initial position or at the final position in the word has significant impact on the pronunciation. The same applies for consonants (e.g. consider the following two names: Adham, Samy)

3.4. Transliteration system for TM evaluation

The transliteration system is built using the moses toolkit [14]. We train a letter-based SMT system on the list of TPs extracted using our TM algorithm explained in section 3.1. The distortion limit is set to 0 to disable any reordering. The transliteration system should be able to learn the proper letter mapping using the alignment of the letters, and hence be able to generate the possible transliterations of a name written in the source language script using the learned mapping rules into a name written in the target language script. This research focuses on the following points:

- Evaluate the performance of TM the algorithm by using the TPs to build a transliteration system. The transliteration system performance is correlated with the quality of the extracted TPs, and hence the TM performance.
- Acquiring a list of target language names for the letter based language model training.
- Study the impact of the segment length on the transliteration quality. In this context, two systems are trained to evaluate the segmentation for the word letters. We compared two segmentation scheme:
  - Simple segmentation of the word by separating individual letters.
  - Advanced segmentation of the word that segment the word to a group of 1-2 letters based on predefined phonetic units which combine two English letters -based on their position in the word- in one substring instead of separate letters (e.g. 'kh', 'kn', 'wh', 'sh' and 'ck').

- The impact of using different tuning metric, we compared the following metrics: TER, BLEU, (TER-BLEU)/2.

3.5. Experiments and evaluation

3.5.1. Purpose and data sets

The objectives of developing our transliteration system is to evaluate the quality of our TM algorithm and perform some research on improving the transliteration quality especially for unseen names in the training data. We evaluated the proposed TM algorithm using Arabic/English parallel corpus which contains about 3.8 million Arabic words and roughly 4.4 million English words. The evaluation of the TM algorithm is performed by training of a statistical system on the extracted TPs and evaluate the quality of transliteration output.

The extracted TPs are divided into three parts:

1. Training data set. The size of the training data is variable based on the selected three levels thresholds (9070 pairs to 10529 TPs).
2. Tuning data set (1k TPs).
3. Test data set. (1k TPs).

All occurrences of words in the TuningSet or TestSet were removed from the training data set.

3.5.2. Evaluation metrics

In order to evaluate the quality of our transliteration system, we used the de-facto standard metrics from ACL Name Entity Workshop (NEWS) [15]: ACC, mean F-Score, MRR, and MAP_{ref}. Here is a short description of each metric:

- ACC=Word Accuracy in Top-1, also known as Word Error Rate. It measures correctness of the first transliteration candidate in the candidate list produced by a transliteration system.
- F-Score= Fuzziness in Top-1. The mean F-score measures how different, on average, the top transliteration candidate is from its closest reference.
- MRR=Mean Reciprocal Rank measures traditional MRR for any right answer produced by the system, among the candidates.
- MAP_{ref} tightly measures the precision in the n-best candidates for the i-th source name, for which reference transliterations are available.

3.5.3. Acquiring a list of target language names for the language model training

We used two resources to get two lists of English names to train our letter based language model (LM). The first resource (LM1) is obtained from the English Gigaword corpus (using only XIN, AFP and NYT parts) by extracting a list of proper names using the Stanford name entity recognizer (NER) [16]. The second resource (LM2) is the English part of the extracted TPs. The Table 1 below compares the results of using LM1 vs. LM2. These results show that the target part (i.e. LM2) of the extracted TPs gives better ACC score while it has some impact on the mean F-score. We decided to use LM2 in all other experiments that measure other variables.

System	ACC	Mean F-Score	MRR	MAP_{ref}
LM1	0.43750	0.88160	0.54787	0.43750
LM2	0.44159	0.87860	0.54862	0.44160

Table 1: *LM1 vs. LM2*

3.5.4. Three levels similarity scores thresholds selections

Several systems were trained to evaluate the best thresholds to be used in our experiments. The experiments show that the best thresholds for 3-scores on tuning set are (TLS_3, TLS_2, TLS_1)=(0, 0.39, 0.49). The thresholds are highly dependent on the normalization functions $Norm_1$, $Norm_2$ and $Norm_3$, so changing the normalization functions will require a re-selection of the three thresholds. The scores of the TuningSet with different thresholds are mentioned in Table 2. Table 3 lists the systems with the TLS scores' thresholds used to select data to train each one.

System(*)	ACC	Mean F-Score	MRR	MAP_{ref}
SYS013 TPs=9167	0.43545	0.87940	0.54188	0.43545
SYS023 TPs=9070	0.44159	0.87860	0.54862	0.44160
SYS034 TPs=10529	0.44774	0.88226	0.55012	0.44774
SYS134 TPs=10529	0.43647	0.88042	0.54220	0.43647

Table 2: *Tuning set results with different thresholds*

System(*)	TLS_3	TLS_2	TLS_1
SYS013	0	0.19	0.39
SYS023	0	0.29	0.39
SYS034	0	0.39	0.49
SYS134	0.19	0.39	0.49

Table 3: *TLS scores' thresholds used for each system*

3.5.5. Segmentations techniques

We used two segmentation techniques, the first technique simply segments the NE into characters, the second one is an

System	ACC	Mean F-Score	MRR	MAP_{ref}
One letter	0.47951	0.89248	0.59226	0.47951
1-2 letters	0.50000	0.89589	0.61178	0.5000

Table 4: *One letter segmentation vs. Advanced segmentation*

advanced segmentation that group together letters that form one phonetic sound in one segment (e.g. ph, ch, sh, etc). Table 4 shows the results of both segmentation techniques. One can see that the second technique helps the letters alignment between source and target and hence improves the transliteration output.

3.5.6. Tuning metric selection

We used the mert tool for weight optimization [17]. We evaluated the impact of using mert tool with different metrics (BLEU, TER and (TER-BLEU)/2. Table 5 shows that (TER-BLEU)/2 gives better results than using BLEU alone or TER alone.

System	ACC	Mean F-Score	MRR	MAP_{ref}
BLEU	0.43648	0.87662	0.54322	0.43647
TER	0.43545	0.87638	0.54263	0.43545
$\frac{(TER-BLEU)}{2}$	0.44159	0.87860	0.54862	0.44159

Table 5: *Experiments with various tuning metrics*

3.5.7. Results

Using three levels similarity scores thresholds=(0, 0.29, 0.39) as explained in section 3.5.4, the total number of extracted TPs is 12988. Table 6 shows the percentage of extracted TPs as a function of the number of aligned words in the parallel text and the number of aligned words with an NNP/NN POS tag.

Data	Number of Words	Extracted TPs %
Bitext-Arabic	3.8M	0.24 %
Bitext-English	4.4M	0.21 %
List of aligned words	1249167	0.73 %
List of aligned NN*	161811	5.6 %

Table 6: *Extracted TPs rate*

In Table 7, we list the transliteration system results using the evaluation metrics mentioned in section 3.5.2. We report the scores for both TuningSet and TestSet. Both TuningSet and TestSet have not seen before in the training data.

System	ACC	Mean F-Score	MRR	MAP_{ref}
TuningSet	0.50000	0.89589	0.61178	0.5000
TestSet	0.46162	0.88412	0.58221	0.4616

Table 7: *TuningSet and TestSet scores*

4. Transliteration mining using comparable corpora - semi-supervised

In this section, we will introduce a corpus based computational method to extract transliteration pairs from comparable corpora. In order to evaluate the extracted pairs, we trained a letter based statistical transliteration system on them and evaluate the system performance which is correlated with the TM quality.

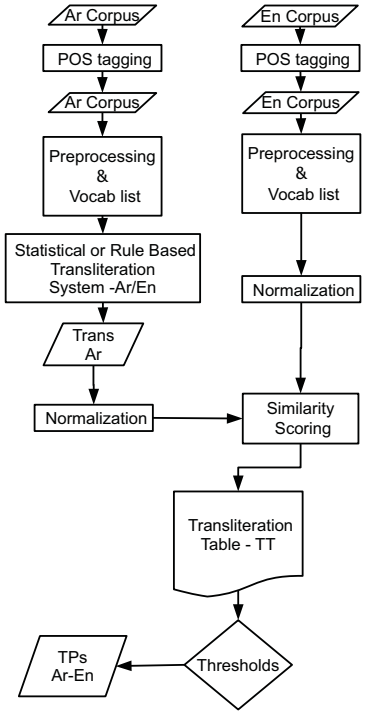

Figure 3: Extracting TPs from comparable corpora

4.1. TM algorithm for comparable corpora

Since it is easy to collect and find monolingual text than parallel text, it would be useful if we can perform TM using this large resources of monolingual text for any pair of languages. This method is inspired by the work of [18] on comparable corpora. We basically do the same at the letter level instead of the word level. Figure 3 shows an overview of the TM algorithm for comparable corpora. The algorithm is designed to remove the non-nouns words in order to minimize

the number of words in each monolingual text, then detects the words which are transliteration of each other, with respect to the observations listed in section 3.3, we score the similarity using three levels similarity scores to generated the transliteration table (TT), which is used later to extract the TPs using three thresholds on the three levels of similarity scores. The following steps explain the TM algorithm:

(1) First, each monolingual corpus is tagged using part-of-speech (POS) tagger. We used Stanford POS tagger [11] for English and Mada/Tokan [12] for Arabic POS tagging.

(2) Then, remove all words with POS tags other than noun (NN) or proper noun (PNN) tags and from the remaining words, remove all English words starts with lower-case letters.

(3) After that removing the POS tags from source text and target text.

(4) Derive two unique words lists (LIST_SRC and LIST_TRG) from both source and target texts.

(5) Then, transliterate source words list (LIST_SRC) into target language (LIST_SRC_TRANS) using rule based transliteration system (or previously created statistical based transliteration system).

(6) Normalize the transliteration of source words list as well as the English words list to the three normalized forms $Norm_1$, $Norm_2$ and $Norm_3$ as explained in section 3.2. The objective of the normalization is folding English letters with similar or close phonetic to same letter or symbol.

(7) Using the normalized values, for each transliterated word in the source language list WORD_AR_TRANS and target language word WORD_EN, calculate the 3-similarity scores between them which are stored in the transliteration table (TT).

(8) Extract TPs from the TT by applying a selected three thresholds on the three levels similarity scores.

4.2. Experiments and evaluation

4.2.1. Purpose and data sets

We evaluated the proposed TM algorithm by applying it on the Arabic Gigaword corpus (about 270.3 million Arabic words using only XIN, AFP and NYT parts) and the English Gigaword corpus (roughly 1470.3 million English words using only XIN, AFP and NYT parts).

We selected the thresholds using empirical method shown in section 4.2.2. The extracted TPs are used as training data. We used the same TuningSet and TestSet extracted from parallel corpus as mentioned in section 3.5.1.

As before, all occurrences of words in the TuningSet or TestSet were removed from the training data.

4.2.2. Three levels similarity scores thresholds selections

Several systems were trained to evaluate the best thresholds to be used in our experiments. Only two thresholds are compared, other thresholds are discarded because they almost give the same TPs. The experiments shows that the best thresholds for 3-scores on tuning set are (TLS_3, TLS_2, TLS_1)=(0, 0.29, 0.39) since they give slightly better mean F-Score and MRR. The scores of the TuningSet with different thresholds are mentioned in Table 8. Table 9 lists the systems with the TLS scores' thresholds used to select data to train each one.

System	ACC	Mean F-Score	MRR	MAP_{ref}
GSYS013 TPs=1.63M	0.30021	0.83973	0.40807	0.30021
GSYS023 TPs=1.96M	0.30021	0.84001	0.40817	0.30021

Table 8: *Tuning set results with different thresholds*

System(*)	TLS_3	TLS_2	TLS_1
GSYS013	0	0.19	0.39
GSYS023	0	0.29	0.39

Table 9: *TLS scores' thresholds used for each system*

4.2.3. Results

Using three levels similarity scores thresholds=(0, 0.29, 0.39) as explained in section 4.2.2, the total number of extracted TPs is 1.96 millions. Table 10 shows TPs rate with respect to the comparable corpora total number of words and the total number of words with NNP/NN POS tag. In Table 11, we list the transliteration system results using the evaluation metrics mentioned in section 3.5.2. We are reporting the scores for both TuningSet and TestSet. Both TuningSet and TestSet has not seen before in the training data.

Data	Number of Words	Extracted TPs %
Arabic Gigaword	270.3 M	0.73%
Arabic Gigaword NN*	18.7 M	10.48%
English Gigaword	1470.3 M	0.13%
English Gigaword NN*	8.1 M	24.20%

Table 10: *Extracted TPs rate*

5. Conclusions

In this paper we introduce a new semi-supervised transliteration mining method for parallel and comparable corpora. The method is mainly based on new suggested Three Levels of Similarity (TLS) scores to extract the transliteration pairs. The transliteration system trained on the transliteration pairs extracted from the parallel corpus achieves an accuracy of 0.50 and a mean F-score of 0.84 on the test set of unseen Arabic names. We also applied our translation mining approach on two Arabic and English monolingual corpora. The system trained on transliteration pairs extracted

System	ACC	Mean F-Score	MRR	MAP_{ref}
TuningSet	0.30021	0.84001	0.40817	0.30021
TestSet	0.27329	0.83345	0.39788	0.27329

Table 11: *TuningSet and TestSet scores*

from comparable corpora achieves an accuracy of 0.30 and a mean F-score of 0.84. This shows that the proposed semi-supervised transliteration mining algorithm is effective and can be applied to other language pairs.

6. Acknowledgment

This research was partially financed by DARPA under the BOLT contract.

7. References

[1] D. Holmes, S. Kashfi, and S. U. Aqeel, "Transliterated arabic name search." in *Communications, Internet, and Information Technology*, M. H. Hamza, Ed. IASTED/ACTA Press, 2004, pp. 267–273.

[2] R. Russell, "Specifications of letters," US patent number 1,261,167, 1918.

[3] K. Darwish, "Transliteration mining with phonetic conflation and iterative training," in *Proceedings of the 2010 Named Entities Workshop*, ser. NEWS '10. Association for Computational Linguistics, 2010, pp. 53–56.

[4] J.-S. Kuo, H. Li, and Y.-K. Yang, "Learning transliteration lexicons from the web," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ser. ACL-44. Association for Computational Linguistics, 2006, pp. 1129–1136.

[5] T. Fukunishi, A. Finch, S. Yamamoto, and E. Sumita, "Using features from a bilingual alignment model in transliteration mining," in *Proceedings of the 3rd Named Entities Workshop (NEWS 2011)*. Chiang Mai, Thailand: Asian Federation of Natural Language Processing, November 2011, pp. 49–57.

[6] A. El-Kahky, K. Darwish, A. S. Aldein, M. A. El-Wahab, A. Hefny, and W. Ammar, "Improved transliteration mining using graph reinforcement," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11. Association for Computational Linguistics, 2011, pp. 1384–1393.

[7] H. Sajjad, A. Fraser, and H. Schmid, "A statistical model for unsupervised and semi-supervised transliteration mining," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, July 2012, pp. 469–477.

[8] Y. Al-Onaizan and K. Knight, "Machine transliteration of names in arabic text," in *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, ser. SEMITIC '02. Association for Computational Linguistics, 2002, pp. 1–13.

[9] S. Jiampojamarn, A. Bhargava, Q. Dou, K. Dwyer, and G. Kondrak, "Directl: a language-independent approach to transliteration," in *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, ser. NEWS '09. Association for Computational Linguistics, 2009, pp. 28–31.

[10] H. Sajjad, A. Fraser, and H. Schmid, "An algorithm for unsupervised transliteration mining with an application to word alignment," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 430–439.

[11] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, ser. NAACL '03. Association for Computational Linguistics, 2003, pp. 173–180.

[12] O. R. Nizar Habash and R. Roth, "Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization," in *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, K. Choukri and B. Maegaard, Eds. Cairo, Egypt: The MEDAR Consortium, April 2009.

[13] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Comput. Linguist.*, vol. 29, no. 1, pp. 19–51, Mar. 2003.

[14] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: open source toolkit for statistical machine translation," in *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ser. ACL '07. Association for Computational Linguistics, 2007, pp. 177–180.

[15] A. K. M. L. Min Zhang, Haizhou Li, Ed., *Report of NEWS 2012 Machine Transliteration Shared Task*, vol. pages 10–20. Jeju, Republic of Korea: Association for Computational Linguistics, July 2012.

[16] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ser. ACL '05. Association for Computational Linguistics, 2005, pp. 363–370.

[17] N. Bertoldi, B. Haddow, and J.-B. Fouet, "Improved minimum error rate training in moses," *Prague Bull. Math. Linguistics*, pp. 7–16, 2009.

[18] S. Abdul Rauf and H. Schwenk, "Parallel sentence generation from comparable corpora for improved smt," *Machine Translation*, vol. 25, no. 4, pp. 341–375, Dec. 2011.