Improving Machine Translation via Triangulation and Transliteration

Nadir Durrani
School of Informatics
University of Edinburgh
dnadir@inf.ed.ac.uk

Philipp Koehn
School of Informatics
University of Edinburgh
pkoehn@inf.ed.ac.uk

Abstract

In this paper we improve Urdu→Hindi⇄English machine translation through triangulation and transliteration. First we built an Urdu→Hindi SMT system by inducing triangulated and transliterated phrase-tables from Urdu–English and Hindi–English phrase translation models. We then use it to translate the Urdu part of the Urdu-English parallel data into Hindi, thus creating an artificial Hindi-English parallel data. Our phrase-translation strategies give an improvement of up to +3.35 BLEU points over a baseline Urdu→Hindi system. The synthesized data improve Hindi→English system by +0.35 and English→Hindi system by +1.0 BLEU points.

1 Introduction

Phrase-based machine translation models are capable of producing translations of reasonable quality but only with large quantities of parallel data. Unfortunately, bilingual data is abundantly available for only a handful of language pairs. The problem of reliably estimating statistical models for translation becomes more of a challenge under sparse data conditions. Previous research has tried to address this bottleneck in two ways i) by making the best use of the existing small corpus by using generalized representations such as morpho-syntactic analysis and suffix separation (Niessen and Ney, 2004; Popović and Ney, 2004; Haque et al., 2012), ii) by generating additional data/inducing models to overcome sparsity (Utiyama and Isahara, 2003; Resnik and Smith, 2003). Techniques like triangulation (Cohn and Lapata, 2007; Wu and Wang, 2007) and paraphrasing (Callison-Burch et al., 2006) have also been used to address the problem of data sparsity. Transliteration is shown to be useful when the languages in question are closely related (Durrani et al., 2010; Nakov and Tiedemann, 2012). Our work falls in this second category of generating additional/data and models.

Hindi and Urdu are widely spoken yet low resourced languages. Hindi descends from Sanskrit and is written in Devanagri script, where as Urdu inherits its vocabulary and language phenomenon from several languages (Arabic, Farsi and Turkish and Sanskrit) and is written in Arabic script. They are a closely related language pair that share grammatical structure and have a high vocabulary overlap.[1] This provides a motivation to build an MT system to create Hindi and Urdu resources by translating one into another.

In this paper, we exploit the relatedness of the two languages and bring together the ideas of triangulation and transliteration to effectively improve Urdu-to-Hindi machine translation. We make use of a tiny Hindi-Urdu parallel corpus, to build a Urdu-to-Hindi translation system. We then improve this system by synthesizing phrase-tables through triangulation and transliteration. We create a triangulated phrase-table using English as a pivot language following the well-known convolution model (Utiyama and Isahara, 2007; Wu and Wang, 2007). The new phrase-table is synthesized using Hindi-English and Urdu-English phrase-tables. We then use the interpolated phrase-table to also synthesize a transliteration phrase-

[1] A small study on 2 newspapers (Dainik Jagran and Hindustan), found that ≈ 40% of the Hindi types overlap with Urdu.

table. We run an EM-based unsupervised transliteration model induction (Durrani et al., 2014c) to extract a list of transliteration pairs from the extracted phrase-table. We use the mined corpus to train a transliteration system. The transliteration system is then used to synthesize a transliteration phrase-table. We produce an n-best list of transliterations for each Urdu word in the tune and test data and create a transliteration phrase table using the features of the transliteration system. The two synthesized phrase-tables are then used as additional set of features along with the regular phrase-table in a log-linear framework. Our integrations give a cumulative improvement of +3.35 BLEU points over the baseline Urdu-to-Hindi system.

In order to demonstrate that the resources in Urdu language can be used for Hindi, we perform an extrinsic evaluation. We use our Urdu-to-Hindi system and translate the Urdu part of the Urdu-English parallel data into Hindi to create an artificial Hindi-English parallel data. Our experiments show that the synthesized parallel data gives an average improvement of +0.35 in Hindi-to-English and +1.0 in English-to-Hindi standard shared task. Our approach is two-way: we use the information in Hindi-English and Urdu-English data to construct a Urdu-to-Hindi system, which we then use for synthesizing Hindi data and subsequently improving Hindi-English translation.

This paper is organized as follows. Section 2 discusses previous efforts on synthesizing parallel data and using pivoting and transliteration to improve MT. Section 3 describe our approach to create interpolated and transliterated phrase-tables and our integration of these into a phrase-based decoder. Section 4 presents the experimental setup and the results. Section 5 concludes the paper.

2 Related Work

There has been a considerable amount of work on synthesizing parallel data and on using triangulation and transliteration to improve machine translation quality. de Gispert and Mariño (2006) induced an English-Catalan parallel corpus by automatically translating Spanish part of English-Spanish parallel data into Catalan with a Spanish-Catalan SMT system. Galuscáková and Bojar (2012) improved English-to-Slovak translation using Czech-English parallel data. Our work is similar to both these efforts except that in their case a lot of parallel data is available for the aiding languages (Czech-English and Spanish-English). In our case both Urdu-English and Hindi-English data are under-resourced. Secondly Urdu and Hindi are written in different scripts so unlike them we need a high quality transliteration module to make use of the common vocabulary. Using pivoting to synthesize phrase-tables has been a widely applied method (Wu and Wang, 2007; Bertoldi et al., 2008; Paul et al., 2009) in SMT. An intermediate language is used to bridge the gap between source and target.

Another approach to alleviate data sparsity is the sentence translation strategy. Rather than building phrase-table source sentences are translated into n pivot sentences which are translated into m target sentences separately. Highest scoring sentences are selected. Utiyama and Isahara (2007) showed that phrase-translation approach is superior to the sentence selection strategy. We will use the phrase-translation strategy to improve the Urdu-to-Hindi translation and sentence selection method to improve Hindi$\rightleftarrows$English translation, although we only use 1-best pivot translation.

A second group of previous research that is related to our work is using transliteration to improve translation for closely related languages. Transliteration has been shown to be useful for more than just translating out-of-vocabulary words and named-entities. Nakov and Tiedemann (2012) built character-based model to improve Bulgarian-Macedonian translation. Durrani et al. (2010) integrated transliteration inside a word-based decoder for Hindi-to-Urdu machine translation. Our work is similar to them, but differs in the following aspects: i) their translation models are based on 1-1/1-N translation links, we do not put any restriction on the alignments ii) their transliteration system is built from hand-crafted rules, our approach is unsupervised and language independent and iii) we additionally integrate pivoting method along with transliteration and demonstrate the usefulness of the synthesized Hindi data. The idea to integrate transliteration module inside of decoder was earlier used by Hermjakob et al. (2008) for the task of disambiguation in Arabic-English machine translation. Much work (Al-Onaizan and Knight, 2002; Zhao et al., 2007) has been done on transliterating named entities and OOVs. Most previous approaches however train a supervised transliteration system separately outside of an MT pipeline, and naïvely replace the OOV words with their 1-best transliterations in the post/pre-processing step

of decoding is commonly used. This work distinguishes from the previous approaches in that it builds a transliteration model automatically from the phrase-tables in an unsupervised fashion and it is language independent.

3 Phrase Translation Strategies

Monolingual data is usually available in a reasonable quantity for many language pairs, but bilingual corpus does not exist or is very sparse. We only need to construct a phrase-table to train a phrase-based SMT system. In this section we will describe our approaches to construct synthetic phrase-tables that help us improve Urdu-to-Hindi translation, and subsequently improve Hindi-to-English and English-to-Hindi translation quality. We used two approaches namely **Triangulation** and **Transliteration** to generate artificial phrase-tables.

3.1 Triangulation

The approach of triangulation is based on using a pivot language to bridge the gap between Urdu and Hindi. Pivot is usually a language closely related to either source or target, or English for which parallel data with either source or target is available in a reasonable quantity. In this work we will use English as a pivot language because we have some Hindi-English and Urdu-English parallel data but very little Urdu-Hindi parallel data. We directly construct Urdu-Hindi phrase-table from Urdu-English and English-Hindi phrase-tables. We train two phrase translation tables $p(\bar{u_i}|\bar{e_i})$ and $p(\bar{e_i}|\bar{h_i})$, using the Urdu-English and English-Hindi bilingual corpora. Given the phrase-table for Urdu-English $p(\bar{u_i}|\bar{e_i})$ and the phrase-table for English-Hindi $p(\bar{e_i}|\bar{h_i})$, we estimate a Urdu-Hindi phrase-table ($p(\bar{u_i}|\bar{h_i})$) using the following model:

$$p(\bar{u_i}|\bar{h_i}) = \sum_{\bar{e_i}} p(\bar{u_i}|\bar{e_i}, \bar{h_i}) p(\bar{e_i}|\bar{h_i})$$

The phrase translation probability $p(\bar{u_i}|\bar{e_i}, \bar{h_i})$ does not depend on the phrase $\bar{h_i}$, because it is estimated from Urdu-English bilingual corpus. The above equation can therefore be rewritten as:

$$p(\bar{u_i}|\bar{h_i}) = \sum_{\bar{e_i}} p(\bar{u_i}|\bar{e_i}) p(\bar{e_i}|\bar{h_i})$$

A phrase-pair $(\bar{u_i}, \bar{h_i})$ is synthesized if there exists an English phrase $\bar{e_i}$ such that $(\bar{u_i}, \bar{e_i})$ exists in the phrase-table $p(\bar{e_i}|\bar{u_i})$ and $(\bar{e_i}, \bar{h_i})$ exists in the phrase-table $p(\bar{h_i}|\bar{e_i})$. The probability of the new phrase-pair is estimated by taking the product of the two and by taking a summation over all such phrases $\bar{e_i}$ for which this condition is true.

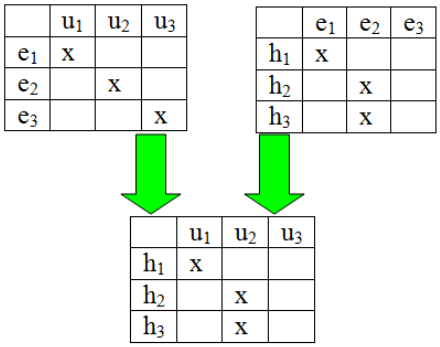

Figure 1: Alignment Induction for Phrase $(\bar{u}, \bar{h})$

Lexical Weighting: Apart from the direct and inverse phrase-translation probabilities, phrase-based translation models also estimate direct and inverse lexical weighting to judge the reliability of a phrase-pair using IBM Model 1. We could estimate these probabilities in the same way as the phrase-translation probabilities as done in Utiyama and Isahara (2007), but we use the more principled *phrase method* as described in Wu and Wang (2007). Given a phrase-pair $(\bar{u}, \bar{h})$ with source words $\bar{u} = u_1, u_2, \ldots, u_n$, target words $\bar{h} = h_1, h_2, \ldots, h_m$ and an alignment a between the source word positions $x = 1, \ldots, n$ and the target word positions $y = 1, \ldots, m$, the lexical feature $p_w(u|e)$ is computed as follows:

$$p_w(\bar{u}|\bar{h}, a) = \prod_{x=1}^{n} \frac{1}{|\{y : (x,y) \in a\}|} \sum_{\forall (x,y) \in a} w(u_x|h_y)$$

But in this scenario we do not have the co-occurring frequencies $c(u_x|h_y)$ to compute $w(u_x|h_y)$. The *phrase method* computes it in the following way: The first step is to extract the alignment information a between the Hindi and Urdu phrases $\bar{h}$ and $\bar{u}$. The alignment information from the phrase-pairs $(\bar{u}, \bar{e})$ and $(\bar{e}, \bar{h})$, can be induced in the following way. Let a_1 and a_2 represent the alignment information inside the phrase-pair $(\bar{u}, \bar{e})$ and $(\bar{e}, \bar{h})$ respectively, then the alignment a between phrase $(\bar{u}, \bar{h})$ can be extracted with the following criteria (See Figure 1 for Example):

$$a = \{(u,h) | \exists e : (u,e) \in a_1 \wedge (e,h) \in a_2\}$$

Given the induced word-alignment a, we can estimate the $w(u|h)$ as follows:

$$w(u|h) = \frac{c(u|h)}{\sum_{u'} c(u'|h)}$$

the co-occurring frequency $c(u|h)$ can be computed with the following criteria:

$$c(u|h) = \sum_{k=1}^{K} p_k(\bar{u}|\bar{h}) \sum_{i=1}^{n} \delta(u, u_i)\delta(h, h_{a_i}))$$

where $\delta(x, y) = 1$ if $x = y$, otherwise $\delta(x, y) = 0$ and $p(\bar{u}|\bar{h})$ is the phrase-translation probability of the k^{th} phrase and K is the total number of phrases synthesized.

3.2 Transliteration

Hindi and Urdu are written in different scripts. A high quality transliteration system is therefore required to capitalize on the vocabulary overlap between Hindi and Urdu. Our second approach attempts to synthesize transliteration phrase-table. A transliteration module can be handy for closely related languages because it can generate novel words that are unknown to the translation corpus but may be justified through the abundantly available language model.

Transliteration Mining: In order to create a transliteration phrase-table, we require a transliteration system and to build such a system we need training data, a list of transliteration pairs for Hindi-Urdu. Such data is not readily available. Instead of creating manually hand-crafted mapping rules for Urdu-to-Hindi transliteration as done in Durrani et al. (2010), we induce a transliteration corpus that we can use to train a character-based SMT system. We induced unsupervised transliteration model (Durrani et al., 2014c) adapting the approach of unsupervised transliteration mining described in (Sajjad et al., 2011, 2012) for the task of machine translation. The algorithm is based on EM. It takes a list of word pairs and extracts transliteration corpus from it. The mining model is a mixture of two components, a transliteration and a non-transliteration sub-model. The overall idea is that the transliteration model ($p_{tr}(h, u)$) tries to maximize the probability of the transliteration pairs in the word-list and the non-transliteration ($p_{ntr}(h, u)$) component tries to fit the rest of the data.

For a Urdu-Hindi word pair (h, u), the **transliteration model** probability for the word pair is defined as follows:

$$p_{tr}(h, u) = \sum_{a \in Align(h,u)} \prod_{j=1}^{|a|} p(q_j)$$

where $Align(e, f)$ is the set of all possible sequences of character alignments, a is one alignment sequence and q_j is a character alignment. The **non-transliteration model** deals with the word pairs that have no character relationship between them. It is modeled by multiplying source and target character unigram models:

$$p_{ntr}(h, u) = \prod_{i=1}^{|h|} p_H(h_i) \prod_{i=1}^{|u|} p_U(u_i)$$

The probabilities of the two models are refined iteratively to extract a list of transliteration corpus. The model is defined as:

$$p(h, u) = (1 - \lambda)p_{tr}(h, u) + \lambda p_{ntr}(u, u)$$

where λ is the prior probability of non-transliteration.

We initially ran mining over Hindi-Urdu parallel data and were able to extract around 2800 transliteration pairs from a word-list of 17000 pairs. Although a transliteration corpus of this size should be enough to build a transliteration model, note that miner's accuracy is not 100%, because of which we also extract pairs that are not transliterations. To extract more transliterations, we additionally feed the interpolated phrase-table ($p(\bar{u}|\bar{h})$), as described above, to the miner. Surprisingly we were able to mine additional 21K transliteration pairs from a list of 95K word pairs.[2]

Transliteration System: Now that we have transliteration word pairs, we can learn a transliteration model. We segment the training corpus into characters and learn a phrase-based system over character pairs. The transliteration model assumes that source and target characters are generated monotonically.[3] Therefore we do not use any reordering models. We use 4 basic phrase-translation features (direct, inverse phrase-translation, and lexical weighting features), language model feature (built from the target-side of mined transliteration corpus), and word and phrase penalties. The feature weights are tuned[4] on a dev-set of 1000 transliteration pairs.

Transliteration Phrase Table: We transliterate tune and test set and extract 100-best transliteration options for each word. We carry forward the 4 translation model features used in the transliteration system to build a transliteration phrase-table.

[2]Miner only uses word pairs with 1-to-1 alignments because M-N/1-N alignments are less likely to be transliterations.

[3]Mining algorithm also makes this assumption.

[4]Tuning data is subtracted from the training corpus while tuning to avoid over-fitting. After the weights are tuned, we add it back, retrain GIZA, and estimate new models.

System	PT	Tune	Test	System	Tune	Test	System	Tune	Test
$\mathbf{B_{u,h}}$	254K	34.01	34.64						
$\bar{\mathbf{B}}_{\mathbf{u,h}}$	254K	34.18	34.79	$\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}$	37.65	37.58			
$\mathbf{T_g}$	10M	15.60	15.34	$\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_r}$	34.77	35.76	$\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}\mathbf{T_r}$	38.0	37.99
$\mathbf{T_r}$		9.54	9.93	$\mathbf{T_g}\mathbf{T_r}$	17.63	18.11		Δ+3.89	Δ+3.35

Table 1: Evaluating Triangulated and Transliterated Phrase-Tables in Urdu-to-Hindi SMT – $\mathbf{B_{u,h}}$ = Baseline Phrase Table, $\bar{\mathbf{B}}_{\mathbf{u,h}}$ = Modified Baseline Phrase Table, $\mathbf{T_g}$ = Triangulated Phrase Table, $\mathbf{T_r}$ = Transliteration Phrase Table

3.3 Integration into Decoder

We simply integrate the synthesized phrase-tables into phrase-based decoder using the standard log-linear approach (Och and Ney, 2004). We search for a Hindi string H which maximizes a linear combination of feature functions:

$$\hat{H} = \arg\max_{H} \left\{ \sum_{j=1}^{J} \lambda_j h_j(U, H) \right\}$$

where λ_j is the weight associated with the feature $h_j(U, H)$.

Overlapping Translation Options: In the default baseline formulation, the three phrase-tables compete with each other during decoding and create a separate decoding path. In some cases however, phrase-tables might agree on the translation options and hypothesize the same translation, in which case they do not have to compete. To avoid such a scenario, and to reward the common translation options, we modify the baseline phrase-table (created from the Hindi-Urdu parallel data) and edit the probabilities for the translation options that exist in the synthesized triangulated and transliterated phrase-tables. For each phrase (u_i, h_i) that exists in the triangulated or transliterated phrase-table we modify its estimates as the following:

$$p_{\bar{b}}(u_i|h_i) = p_b(u_i|h_i) + p_{tg}(u_i|h_i) + p_{tr}(u_i|h_i)$$
$$\forall (u_i, h_i) | (u_i, h_i) \in p_{tg}(u|h) \vee (u, h) \in p_{tr}(u_i|h_i)$$

where $p_b(u|h)$ is the baseline phrase-table, $p_{tg}(u|h)$ is the triangulated phrase-table and $p_{tr}(u|h)$ is the transliterated phrase-table. This modification slightly improves the performance of the baseline system.

LM-OOV Feature: A lot of the words that the transliteration model produces would be unknown to the language model. To create a bias towards the transliteration options that are known to the language model, we additionally use an LM-OOV feature which counts the number of words in a hypothesis that are unknown to the language model. Language model feature alone can not handle the unknown transliterations, because the smoothing methods such as Kneser-Ney assign significant probability mass to unseen events, thus giving high probability to such unknown transliterations. The LM-OOV feature acts as a prior to penalize such hypotheses. The feature is tuned along with the regular features. Therefore if such transliterations are useful, the optimizer can assign positive weight to this feature. But we noticed that optimizer assigned a high negative weight to this feature, thus heavily penalizing the unknown words.

4 Evaluation

4.1 Data

Our baseline Urdu-to-Hindi system is built using a small EMILLE corpus (Baker et al., 2002) which contain roughly 12000 sentences of Hindi and Urdu sentences which are not exactly parallel. After sentence alignment, we were able to extract a little more than 7000 sentence pairs. The model for Urdu-English data was build using Urdu-English segment of the Indic[5] multi-parallel corpus (Post et al., 2012) which contain roughly 87K sentences. The Hindi-English systems were trained using Hindi-English parallel data (Bojar et al., 2014) composed by compiling several sources including the Hindi-English segment of the Indic parallel corpus. It contains roughly 273K parallel sentences. The tune and test sets for Hindi-Urdu task were created by randomly selecting 1800 sentences from the EMILLE corpus which were then removed from the training data to avoid over-fitting. We use half of the selected sentences for tuning and other half for test. The dev and test sets for Hindi-English translation task are the standard sets news-dev2014 and news-

[5]The multi-indic parallel corpus also have Hindi-English segment, but the data is completely disjoint from the Urdu-English segment.

test2014 containing 1040 and 2507 sentences respectively. We split news-dev2014 into two halves and used the first half for tuning and second as a test along with the official news-test2014 set. To get more stabilized tuning weights, the tune set is concatenated with Hindi-English dev-set (1400 Sentences) made available with the Hindi-English segment of the Indic parallel corpus. We trained the language model using all the English (287.3M Sentences) and Hindi (43.4M Sentences) monolingual data made available for the 9^{th} Workshop of Statistical Machine Translation.

4.2 Baseline System

Urdu-to-Hindi: We trained a phrase-based Moses system with the following settings: A maximum sentence length of 80, GDFA symmetrization of GIZA++ alignments, an interpolated Kneser-Ney smoothed 5-gram language model with KenLM (Heafield, 2011) used at runtime, 100-best translation options, MBR decoding (Kumar and Byrne, 2004), Cube Pruning (Huang and Chiang, 2007), using a stack size of 1000 during tuning and 5000 during test. We tuned with the k-best batch MIRA (Cherry and Foster, 2012). Because Hindi and Urdu have same grammatical structure, we used a distortion limit of 0 and no reordering.[6]

Hindi-English: For Hindi-English systems, we additionally used hierarchical lexicalized reordering (Galley and Manning, 2008), a 5-gram OSM, (Durrani et al., 2013), sparse lexical and domain features, (Hasler et al., 2012), class-based models (Durrani et al., 2014b), a distortion limit of 6, and the no-reordering-over-punctuation heuristic.

4.3 Experiments

Urdu-to-Hindi: In the initial experiments, we evaluated the effect of integrating the synthesized phrase-tables into Urdu-to-Hindi machine translation. Table 1 shows results on our Urdu-to-Hindi system. Our modification ($\bar{\mathbf{B}}_{\mathbf{u,h}}$) to the baseline phrase-table ($\mathbf{B}_{\mathbf{u,h}}$) to reward the translation options common between the phrase-tables improve the performance of the baseline system slightly (+0.15). Both triangulated ($\mathbf{T_g}$) and transliterated ($\mathbf{T_r}$) phrase-tables show value when used in isolation, although their performance (BLEU (Papineni et al., 2002)) is a lot worse in comparison to the baseline. Some of this difference in performance can be attributed to the fact that the tune and test sets used for evaluation were extracted from the training data. The real difference of performance can not be studied without a dedicated Urdu-Hindi test set which unfortunately is not available. However, our Hindi-English evaluation shows that this speculation is correct (See below for further discussion). Adding triangulated phrase-table ($\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}$) to the modified baseline system gives an improvement of +2.79. In comparison, adding transliteration phrase-table ($\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_r}$) gives an improvement of +0.97. An overall improvement of +3.35 is observed when all three phrase-tables ($\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}\mathbf{T_r}$) are used together.

Hindi-English: We carried out an extrinsic evaluation to measure the quality of our Urdu-to-Hindi translation systems. We translated the Urdu part of the Urdu-English parallel data using the Urdu-to-Hindi SMT systems described above. We then used the translated corpus to from a synthetic Hindi-English parallel corpus and evaluated its performance by adding it to the baseline Hindi-English systems and in isolation. Table 2 shows the results on adding the synthesized Hindi-English parallel data on top of competitive Hindi-English baseline systems and in isolation. The system ($\mathbf{B}_{\mathbf{h,e}}\mathbf{D}_{\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}\mathbf{T_r}}$) which uses the data generated from our best Urdu-to-Hindi system ($\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}\mathbf{T_r}$) gives an average improvement of +0.35 BLEU points on Hindi-to-English translation task and +1.0 BLEU points on English-to-Hindi. The performance of the system built using synthesized data only ($\mathbf{D}_{\bar{\mathbf{B}}_{\mathbf{u,h}}\mathbf{T_g}\mathbf{T_r}}$) is significantly worse than the baseline system on Hindi-to-English task, however the difference is not as much in the other direction. We believe this is still a positive result given the fact that our data is artificially created and is three times smaller than the Hindi-English parallel data used to build the baseline system.

We also synthesized data using the baseline system only trained on the EMILLE corpus ($\bar{\mathbf{B}}_{\mathbf{u,h}}$) and using synthesized phrase-tables ($\mathbf{T_g}\mathbf{T_r}$) separately. The results in row $\bar{\mathbf{B}}_{\mathbf{h,e}}\mathbf{D}_{\mathbf{B}_{\mathbf{u,h}}}$ shows that the data synthesized from the baseline Urdu-Hindi system ($\bar{\mathbf{B}}_{\mathbf{u,h}}$) is harmful in both the Hindi-English tasks. In comparison the data synthesized from triangulated and transliterated Urdu-to-Hindi system still showed an average improvement of +0.65 in English-to-Hindi task. No gains were observed in the other direction. Doing an error anal-

[6]Results do not improve with reordering enabled.

System	Hindi-to-English		English-to-Hindi	
	new-dev$_{14}$	news-test$_{14}$	new-dev$_{14}$	news-test$_{14}$
Baseline ($\mathbf{B_{h,e}}$)	17.11	15.77	11.74	11.57
$\mathbf{B_{h,e}D_{\bar{B}_{u,h}T_gT_r}}$	17.60 Δ+0.49	15.97 Δ+0.20	12.83 Δ+1.09	12.47 Δ+0.90
$\mathbf{D_{\bar{B}_{u,h}T_gT_r}}$	13.13 Δ-3.98	10.96 Δ-4.79	11.14 Δ-0.60	10.51 Δ-1.06
$\mathbf{B_{h,e}D_{\bar{B}_{u,h}}}$	16.91 Δ-0.20	15.39 Δ-0.36	10.63 Δ-1.11	9.87 Δ-1.7
$\mathbf{B_{h,e}D_{T_gT_r}}$	17.15 Δ+0.04	15.84 Δ+0.10	12.47 Δ+0.73	12.13 Δ+0.56

Table 2: Evaluating Synthesized Hindi-English Parallel Data on Standard Translation Task – $\mathbf{D_{\bar{B}_{u,h}T_gT_r}}$ = System using data synthesized from the best Urdu-to-Hindi System that additionally use Triangulated and Transliterated Phrase Tables

ysis we found that the baseline Urdu-Hindi system suffers from data sparsity. The number of out-of-vocabulary tokens when translating the Urdu corpus using baseline phrase-table were 310K. In comparison the number of words unknown to the interpolated phrase-table were 50K and these were translated using in-decoding transliteration method (Durrani et al., 2014c).[7]

5 Conclusion

In this paper we applied a combination of pivoting and transliteration to improve Urdu→Hindi⇄English machine translation using a two-way approach. First we use the Urdu-English and English-Hindi phrase-tables to induce a Urdu-to-Hindi translation model. We then use the resultant model to synthesize additional Hindi-English parallel data. Both transliteration and triangulated phrase-tables showed improvements over the baseline system. A cumulative improvement of +3.35 BLEU points was obtained using both in tandem. The artificially induced parallel data gives an improvement of +0.35 for Hindi-to-English and +1.0 for English-to-Hindi over a competitive baseline system. Our English-to-Hindi system was ranked highest for EN-HI and third for HI-EN in this year's WMT translation task (Durrani et al., 2014a).

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements n ° 287658 (EU-Bridge) and n ° 287688 (MateCat).

[7]Note that we also applied transliteration for the words that were known to the interpolated phrase-table.

References

Al-Onaizan, Y. and Knight, K. (2002). Translating Named Entities Using Monolingual and Bilingual Resources. In *Proceedings of ACL*, pages 400–408.

Baker, P., Hardie, A., McEnery, T., Cunningham, H., and Gaizauskas, R. J. (2002). EMILLE, a 67-million word corpus of indic languages: Data collection, mark-up and harmonisation. In *LREC*.

Bertoldi, N., Barbaiani, M., Federico, M., and Cattoni, R. (2008). Phrase-based Statistical Machine Translation with Pivot Languages. *Proceeding of IWSLT*, pages 143–149.

Bojar, O., Diatka, V., Rychlý, P., Straňák, P., Tamchyna, A., and Zeman, D. (2014). Hindi-English and Hindi-only Corpus for Machine Translation. In *Proceedings of the Ninth International Language Resources and Evaluation Conference (LREC'14)*, Reykjavik, Iceland. ELRA, European Language Resources Association. in prep.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24, New York City, USA. Association for Computational Linguistics.

Cherry, C. and Foster, G. (2012). Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada. Association for Computational Linguistics.

Cohn, T. and Lapata, M. (2007). Machine Translation by Triangulation: Making Effective Use of Multi-Parallel Corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735, Prague, Czech Republic. Association for Computational Linguistics.

de Gispert, A. and Mariño, J. B. (2006). Statistical Machine Translation without Parallel Corpus: Bridging through Spanish. In *In Proceedings of LREC 5th Workshop on Strategies for developing Machine Translation for Minority Languages*, pages 65–68.

Durrani, N., Fraser, A., Schmid, H., Hoang, H., and Koehn, P. (2013). Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria. Association for Computational Linguistics.

Durrani, N., Haddow, B., Koehn, P., and Heafield, K. (2014a). Edinburgh's Phrase-based Machine Translation Systems for WMT-14. In *Proceedings of the ACL 2014 Ninth Workshop on Statistical Machine Translation*, Baltimore, MD, USA.

Durrani, N., Koehn, P., Schmid, H., and Fraser, A. (2014b). Investigating the Usefulness of Generalized Word Representations in SMT. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, Dublin, Ireland.

Durrani, N., Sajjad, H., Fraser, A., and Schmid, H. (2010). Hindi-to-Urdu Machine Translation through Transliteration. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 465–474, Uppsala, Sweden. Association for Computational Linguistics.

Durrani, N., Sajjad, H., Hoang, H., and Koehn, P. (2014c). Integrating an Unsupervised Transliteration Model into Statistical Machine Translation. In *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL 2014)*, Gothenburg, Sweden. Association for Computational Linguistics.

Galley, M. and Manning, C. D. (2008). A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii.

Galuscáková, P. and Bojar, O. (2012). Improving SMT by Using Parallel Data of a Closely Related Language. In *Baltic HLT*, pages 58–65.

Haque, R., Penkale, S., Jiang, J., and Way, A. (2012). Source-side suffix stripping for bengali-to-english smt. In *IALP*, pages 193–196.

Hasler, E., Haddow, B., and Koehn, P. (2012). Sparse Lexicalised features and Topic Adaptation for SMT. In *Proceedings of the seventh International Workshop on Spoken Language Translation (IWSLT)*, pages 268–275.

Heafield, K. (2011). KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom.

Hermjakob, U., Knight, K., and Daumé III, H. (2008). Name Translation in Statistical Machine Translation - Learning When to Transliterate. Columbus, Ohio.

Huang, L. and Chiang, D. (2007). Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Kumar, S. and Byrne, W. J. (2004). Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *HLT-NAACL*, pages 169–176.

Nakov, P. and Tiedemann, J. (2012). Combining Word-Level and Character-Level Models for Machine Translation Between Closely-Related Languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305, Jeju Island, Korea. Association for Computational Linguistics.

Niessen, S. and Ney, H. (2004). Statistical Machine Translation with Scarce Resources using Morpho-Syntactic Information. *Computational Linguistics*, 30(2):181–204.

Och, F. J. and Ney, H. (2004). The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(1):417–449.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Morristown, NJ, USA.

Paul, M., Yamamoto, H., Sumita, E., and Nakamura, S. (2009). On the Importance of Pivot Language Selection for Statistical Machine Translation. In *HLT-NAACL (Short Papers)*, pages 221–224. The Association for Computational Linguistics.

Popović, M. and Ney, H. (2004). Towards the use of word stems and suffixes for statistical machine translation. In *Proceedings of COLING*, Geneva, Switzerland.

Post, M., Callison-Burch, C., and Osborne, M. (2012). Constructing Parallel Corpora for Six Indian Languages via Crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada. Association for Computational Linguistics.

Resnik, P. and Smith, N. A. (2003). The Web As a Parallel Corpus. *Comput. Linguist.*, 29(3):349–380.

Sajjad, H., Durrani, N., Schmid, H., and Fraser, A. (2011). Comparing two techniques for learning transliteration models using a parallel corpus. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 129–137, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Sajjad, H., Fraser, A., and Schmid, H. (2012). A statistical model for unsupervised and semi-supervised transliteration mining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, Jeju, Korea.

Utiyama, M. and Isahara, H. (2003). Reliable Measures for Aligning Japanese-English News Articles and Sentences. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 72–79, Sapporo, Japan. Association for Computational Linguistics.

Utiyama, M. and Isahara, H. (2007). A Comparison of Pivot Methods for Phrase-Based Statistical Machine Translation. In *2007 Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 484–491.

Wu, H. and Wang, H. (2007). Pivot Language Approach for Phrase-Based Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 856–863, Prague, Czech Republic. Association for Computational Linguistics.

Zhao, B., Bach, N., Lane, I., and Vogel, S. (2007). A Log-Linear Block Transliteration Model based on Bi-Stream HMMs. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, New York.