Raj Nath Patel*, Prakash B. Pimpale and M. Sasikumar

Machine Translation in Indian Languages: Challenges and Resolution

https://doi.org/10.1515/jisys-2018-0014
Received January 8, 2018; previously published online September 4, 2018.

Abstract: English to Indian language machine translation poses the challenge of structural and morphological divergence. This paper describes English to Indian language statistical machine translation using preordering and suffix separation. The preordering uses rules to transfer the structure of the source sentences prior to training and translation. This syntactic restructuring helps statistical machine translation to tackle the structural divergence and hence provides better translation quality. The suffix separation is used to tackle the morphological divergence between English and highly agglutinative Indian languages. We demonstrate that the use of preordering and suffix separation helps in improving the quality of English to Indian language machine translation.

Keywords: Statistical machine translation, reordering, suffix and compound splitting, transliteration.

Classification: 68T50.

1 Introduction

In this paper, we present our statistical machine translation (SMT) experiments from English to Tamil, Malayalam, Punjabi, and Hindi. From the set of target languages involved, Hindi and Punjabi belong to the Indo-Aryan language family, whereas Tamil and Malayalam belong to the Dravidian language family. All languages except English have the same flexibility toward word order, canonically following the subject-object-verb structure, whereas English follows the subject-verb-object structure.

The structural difference between the source and target language makes SMT difficult. It has been demonstrated that preordering benefits SMT in such cases [36, 38]. Preordering or reordering transforms the source sentence into a target-like order using the syntactic parse tree of the source text. After reordering, training of the SMT system is performed using parallel corpus. Reordering also applies to the new source sentences prior to decoding. The reordering system is generally developed using a rich set of rules. These rules are manually extracted based on analysis of parsed source sentence and corresponding target language translation. We used the reordering system developed by Patel et al. [25].

With reference to the morphology, Tamil and Malayalam are more agglutinative compared to English. It is also known that SMT produces more unknown words, resulting in bad translation quality if the morphological divergence between the source and the target language is high. Previous researchers [14, 15, 28, 29, 33] have demonstrated ways to handle this issue with morphological segmentation before training the SMT system. To tackle the morphological divergence between English and Tamil, we used a suffix separation system developed by Patel et al. [26] and Pimpale et al. [28] as a preprocessing step. The suffix separation tries to reduce the morphological divergence between the source and agglutinative target language by splitting compound words. Such words when generated in decoding are combined to form a single word using a postprocessor.

A factored SMT with the stem as an alignment factor [13] has been trained to achieve better alignment. The target side transliteration is also applied to non-translated words.

***Corresponding author: Raj Nath Patel,** KBCS Division, Centre for Development of Advanced Computing, Gulmohar Cross Road No. 9, Opp Juhu Shopping Center, Juhu, Mumbai 400049, India, e-mail: patelrajnath@gmail.com
Prakash B. Pimpale and M. Sasikumar: KBCS Division, Centre for Development of Advanced Computing, Gulmohar Cross Road No. 9, Opp Juhu Shopping Center, Juhu, Mumbai 400049, India

The rest of the paper is organized as follows. In Section 2, we discuss the challenges of English to Indian language machine translation (MT). Section 3 describes the dataset and experimental setup. Section 4 discusses experiments and results, followed by a description of our submission to the shared task in Section 5. In Section 6, we report the few early observations. Finally, conclusion and future work are discussed in Section 7.

2 Challenges of English to Indian Language SMT

As discussed briefly in Section 1, English to Indian language MT poses the challenges of structural and morphological differences. In the following subsections, we discuss the syntactic and morphological divergences with examples.

2.1 Syntactic Divergence

The important structural difference in English and most of the Indian languages is the word order. English uses the subject-verb-object order, whereas most of the Indian languages, including the ones under study, primarily use subject-object-verb. Some of the Indian languages are of the nature of free word order. Prepositions in case of English come after the pronoun or noun they qualify; for Hindi, they succeed the noun or pronouns, also known as postpositions. Two representative examples are given in Table 1. In the first example, we can see that word order "ate mango" becomes "mango ate" [*aama khaayaa*; all non-English (Hindi, Tamil) words have been written in Itrans using http://sanskritlibrary.org/transcodeText.html; for Tamil, we have written the word pronunciation in Devanagari and then trans-coded in Itrans] in Hindi. In the second example, the preposition "on" (*para* in Hindi) becomes the postposition of the noun phrase "the table" (*tebala*).

2.2 Morphological Divergence

We discuss here the morphological divergence of Tamil and Malayalam with respect to English using analysis based on the parallel corpus detailed in Table 2. The purpose behind comparing Tamil and Malayalam with English was to demonstrate the difference of agglutination. The morphological divergence for Hindi and Punjabi with respect to English would not be as high as these languages. In our old studies [27, 28], we have compared Marathi, Tamil, Telugu, and Bengali with Hindi, where all the languages were more agglutinative as compared to Hindi.

We know that the parallel corpus represents the same information in two different languages. In Table 2, we can see that English makes use of more words to represent the same concept or information as compared to Tamil and Malayalam. If we look at the unique words for each language, we can conclude that English has

Table 1: Example of Different Word Orders in English and Hindi.

English sentence	Hindi sentence
Ram ate mango	*raama ne* (Ram) *aama* (mango) *khaayaa* (ate)
Apple is on the table	*seba* (Apple) *tebala* (the table) *para* (on) *hai* (is)

Table 2: Statistical Analysis of Morphological Divergence.

	English-Malayalam		English-Tamil	
	English	Malayalam	English	Tamil
#Sentences	103 K	103 K	139 K	139 K
#Total words	1673 K	1069 K	2189 K	1576 K
#Unique words	51 K	209 K	71 K	255 K
Average word length (#characters)	8.02	12.40	8.12	11.95
Average sentence length (#words)	16.31	10.42	15.75	11.33

much less vocabulary as compared to these two Indian languages. This implies that English needs to make use of different combinations of available words to represent various concepts, whereas in Tamil and Malayalam, different concepts are represented by different words. Examples of the same can be seen in Table 3. The average sentence length of these languages also establishes the same fact. The significant difference in average word length shows that the words of Tamil and Malayalam are longer as compared to that of English. Many new words in these Indian languages are formed by compounding of words or suffixes. The phenomenon is called agglutination, and so we say that Tamil and Malayalam are more agglutinative than English.

The difference in the length of source and target sentence makes the word alignment difficult. The wrong alignment ultimately results in poor-quality translation. In our experiments, we try to tackle this issue by using suffix separation methods for English-Tamil SMT.

3 System Setup

In the following subsections, we describe the data distribution followed by preprocessing, evaluation metrics, and SMT system setup used for the experiments.

3.1 Data Set

For our experiments, we used the corpus shared by MTIL-2017 [35], detailed in Table 4. We split the shared data into train, test, and development sets. We used the publicly available *Indian language tokenizer and text normalizer* for all the target languages. For English, we used the tokenizer available with *moses* (https://github.com/moses-smt/mosesdecoder). For long sentences, the expectation-maximization algorithm has a hard time learning the word alignments. Also, if the source-to-target word length ratio is very high, it implies misaligned segments. Thus, we removed the sentences having word count >80 or source-target word length ratio $>1{:}9$.

3.2 Preprocessing

SMT works well if the structural divergence between source and target language is not very high. To reduce the structural divergence between source and target language, we used source side reordering. To tackle the morphological divergence between the source and target, we preprocessed the Tamil with suffix separation.

Reordering is a preprocessing stage for SMT system where the words of the source sentence are restructured as per the syntax of the target language prior to training. The test set is also preprocessed similar to the training data prior to decoding. The idea is to facilitate the training process by better alignments and parallel phrase extraction for a phrase-based SMT system. Reordering also helps the decoding process and hence

Table 3: Example English Phrases and Equivalent Tamil Words.

English	Tamil
Have to go	*pokanuma*
That too	*aTavuma*

Table 4: Data Distribution.

	Training		Development		Testing	
	#Sents	#Words	#Sents	#Words	#Sents	#Words
English-Malayalam	101 K	1846 K	500	9134	500	9450
English-Hindi	159 K	2954 K	500	8891	500	9168
English-Punjabi	128 K	2089 K	500	8247	500	8337
English-Tamil	138 K	2442 K	500	8833	500	8475

improves the MT quality. A detailed analysis of reordering, improving the training, and translation quality was done by Gupta et al. [9].

For English-Hindi SMT, earlier reordering was used [25, 36, 38] and have shown significant improvements over baseline. Kunchukuttan et al. [17] reported SMT results for English to 10 major Indian languages, and showed that reordering helps for all of them.

Other language pairs have also shown significant improvement when reordering is employed. Xia and McCord [41] and Wang et al. [40] have observed improvement for French-English and Chinese-English language pairs, respectively. Nießen and Ney [22] have proposed sentence restructuring, whereas Collins et al. [4] have proposed clause restructuring to improve German-English SMT. Popovic and Ney [30, 31] have also reported the use of simple local transformation rules for Spanish-English and Serbian-English translation. Recently, Khalilov and Fonollosa [11] proposed a reordering technique using a deterministic approach for long distance reordering and non-deterministic approach for short distance reordering exploiting morphological information. Some reordering approaches are also presented exploiting the SMT itself [5, 9].

Suffix separation is the process where the words are split into stem and suffixes. For MT, the splitting of an unknown word into its parts enables the translation of the word by the translation of its parts. For example (Hindi-Marathi SMT), in Marathi, "*mahinyaaMnii*" is translated as "*mahiine meM*" (in the month) in Hindi. In this case, we split the word into "*mahiny* + *aaMnii*." Here, the suffix "*aaMnii*" corresponds to the word "*meM*" in Hindi.

We considered only suffixes from the target language (Tamil), which correspond to prepositions in the source language (English). For this task, the list of suffixes (#suffixes = 16) is manually created with the linguistic expertise. When a word is subjected to suffix separation, the longest matching suffix from the list is considered for the suffix separation. The suffix separation takes place only once for a word. We add a continuation symbol "@@" after the stem word (mahiny@@), which is used to combine the suffixes back after translation. The pseudo-code for the suffix separation is detailed in Algorithm 1.

Algorithm 1: Suffix Separation.

```
1: procedure SuffixSep(word)
2:     suffixSet ← read file suffix list
3:     splits ← {word, "NULL"}
4:     for suffix ← suffixSet do
5:         if then word.ENDSWITH = suffix & word.LENGTH > suffix.LENGTH
6:             splits[0] ← word.SUBSTRING(0, word.LASTINDEXOF(suffix)) + "@@"
7:             splits[1] ← suffix return splits
8:         end if
9:     end for
10: end procedure
```

Many researchers have tried compound word splitting and suffix separation for SMT between morphologically rich languages. Brown [1] has proposed an approach guided by a parallel corpus. The work is limited to breaking compounds into cognates and words found in a translation lexicon, but no results on translation performance are reported. Koehn and Knight [12] have demonstrated an empirical method of learning the compound splitting using monolingual and bilingual data and reported impact on performance of SMT. Patel et al. [26] and Pimpale et al. [28] reported significantly improved translation quality for Indian language SMT using suffix separation and compound word splitting.

3.3 Transliteration

Out-of-vocabulary (OOV) words occur in almost all MT systems. These words are mostly named entities, technical terms, or foreign words that were not part of the training corpus or were not added to the development dictionary. Therefore, OOV words need to be translated to the target language using transliteration. Transliteration helps improve the translation quality [26], and it has also been shown to be useful for translating

closely related language pairs [6, 21]. For most of the language pairs, a parallel corpus of transliterations is not readily available. Moreover, even if such training data are made available, the arrangement to integrate transliterated words into MT pipelines are not available in SMT toolkits like phrasal [8] and joshua [34].

Generally, a transliteration system is trained separately outside of an MT pipeline using supervised training methods. It gives all possible target transliterations for a given source word. Generally, the one-best output is selected as transliteration and is used to replace the OOV word in the translation, after decoding.

This paper uses an unsupervised model [7] based on the expectation-maximization to induce transliteration corpus using parallel data, which is then used to train a transliteration model. The implementation is available with the *moses* toolkit. We used the top 100-best transliteration output for OOV words. These candidates are plugged in the translation replacing OOV words and rescored with the language model to get the best translation for source sentence.

3.4 SMT System Setup

The baseline system was setup by using the phrase-based model [2, 14, 19, 23]. Koehn and Hoang [13] used the factored model. We tuned the model parameters using minimum error rate training [23]. The language model was trained using the KenLM [10] toolkit with modified Kneser-Ney smoothing [3]. We tried various n-gram language models and found that 5-gram performs best for the languages under study. For factored SMT training source and target side, stem has been used as an alignment factor. Stemming for Hindi, Punjabi, Tamil, and Malayalam has been done using a modified version of lightweight stemmer [37]. For English, we have used porter stemmer [20].

3.5 Evaluation Metrics

The different experimental systems are being compared using BLEU [24], PER [32], TER [39], and CDER [18]. For an MT system to be better, higher BLEU scores with lower PER, TER, and CDER are desired.

4 Experiments and Results

We carried out various experiments to achieve better accuracy, using the data and system setup described in previous sections. Table 5 details the experiments we tried. We report BLEU, 1-TER, 1-PER, and 1-CDER for the

Table 5: Translation Quality Scores for Different Systems.

		BLEU	1-TER	1-PER	1-CDER
English-Malayalam	S1	08.52	13.63	32.32	21.46
	S2	08.15	14.37	32.74	21.57
	S3	08.10	09.85	24.07	20.36
	S4	08.25	10.03	24.38	20.52
English-Hindi	S1	16.75	27.05	51.73	33.95
	S2	18.74	31.30	51.94	37.37
	S3	19.30	33.38	52.35	37.61
	S4	19.43	33.53	52.57	37.77
English-Punjabi	S1	21.71	38.26	56.13	41.44
	S2	23.09	40.90	56.83	44.06
	S3	22.17	39.20	56.25	42.77
	S4	22.26	39.35	56.48	42.88
English-Tamil	S1	06.20	13.05	32.72	21.97
	S2	07.44	16.35	32.29	24.43
	S3′	07.47	17.87	34.86	23.49
	S4′	07.56	18.01	35.06	23.62

S1: BL; S2: BL + RO; S3: BL + RO + FACT; S3′: BL + RO + SPLIT + FACT; S4: BL + RO + FACT + TR; S4′: BL + RO + SPLIT + FACT + TR; BL: baseline; RO: reordering; FACT: factored models; TR: transliteration.

Table 6: Comparison of Translation with an Example of English-Hindi SMT.

Source	Ahmedabad was named after the sultan Ahmed Shah, who built the city in 1411. (English)
S1	Ahmedabad was named after the sultan Ahmed Shah, who built the city in 1411. (English) *ahamadaabaada ke naama para rakhaa gayaa sultaana ahamada shaaha vaale shahara 1411.* (machine-translated Hindi)
S3	Ahmedabad the sultan Ahmed Shah after named was, who 1411 in the city built. (reordered English) *ahamadaabaada kaa naama sultaana ahamadashaaha ke naama se paDaa thaa jisane 1411 meM shahara banavaayaa thaa.* (machine-translated Hindi)
Reference	*ahamadaabaada kaa naama sultaana ahamadashaaha ke naama para paDaa thaa, jisane 1411 meM shahara banavaayaa thaa.* (manually translated Hindi)

various experiments. TER, PER, and CDER are the word error rates (WERs) to measure the quality of translation. In general, these scores should be low for a better MT system. In contrast, 1-WER implies that the higher is the value, the better would be the accuracy. It can be seen that the use of preprocessing and transliteration has contributed to the improvement of 1 to 1.5 BLEU points over the baseline for English-Hindi, English-Punjabi, and English-Tamil. For English-Malayalam, the BLEU has decreased and we plan to investigate this in our future work. Also, investigation is needed to figure out why the BLEU score decreased on use of factors in English-Punjabi, while it was useful for other language pairs.

Table 6 describes with an example how reordering reduces the structural divergence and helps achieve better translation quality. From the example, it can be seen that the translation of the system using S3 is better than S1. The output of S3 is structurally more correct and conveys the same meaning as that of the reference translation.

5 Submission to the Shared Task

As shown in Table 7, we (C-DACM) submitted our systems for all the language pairs in the shared task. The submitted translations, of the unseen test set, were obtained using S4 and S4′. The submitted systems were manually evaluated by three native speakers for adequacy, fluency, and rating. The average of the three manual evaluators is given in the Table 5. The shared task organizers (MTIL17) used the percentage of adequacy and fluency as the primary metric for the shared task. Evaluation results for the top three participating systems were published by MTIL17 [16], as shown in Table 7. From the evaluation results, it is evident that our (CDAC-M) submissions significantly outperform the other submissions for English-Hindi, English-Tamil, and English-Malayalam. For English-Punjabi, our submission stands in the second position.

Table 7: Submissions at MTIL2017.

Languages	Team	Avg adequacy (A)	Avg fluency (F)	Avg rating (R)	A&F%	R%	BLEU
English-Malayalam	CDAC-M	1.92	1.67	1.60	38.34	31.94	2.60
English-Hindi	CDAC-M	3.82	3.63	3.43	74.53	68.64	20.64
	NIT-M	3.27	3.56	3.26	68.27	65.14	23.25
	IIT-B	2.55	3.23	2.59	57.81	51.87	21.01
	JU	1.81	1.72	1.58	35.28	31.50	3.57
English-Punjabi	NIT-M	3.38	3.74	3.235	67.55	65.05	9.24
	CDAC-M	3.05	3.02	2.92	60.91	58.34	8.68
	IIT-B	2.65	2.71	2.62	52.93	52.4	11.38
English-Tamil	CDAC-M	2.61	2.57	2.40	52.26	48.00	6.15
	HANS	2.16	2.12	2.17	43.22	43.50	1.93
	NIT-M	1.59	1.65	1.58	31.72	31.74	1.31

CDAC-M, Centre for Development of Advanced Computing, Mumbai, India; IIT-B, Indian Institute of Technology, Bombay, India; NIT-M, National Institute of Technology, Mizoram, India; JU, Jadavpur University, Kolkata, West Bengal, India; HANS, New York University, New York City, NY, United States; Avg, average of three manual evaluation scores.

Table 8: Reordering Errors.

English	**Certain foods are very useful for losing fat.**
Reordered	Certain foods very useful fat losing for are.
System output	*kuCha khaadya padaarthoM ko khone ke lie bahuta upayogii phaiTa hote haiM*
Expected reordering	Certain foods fats losing for very useful are.
Expected output	*kuCha khaadya padaartha vasaa khone ke lie bahuta upayogii hote hai*.

6 Error Analysis

A closer look at the performance of these systems to understand the utility of reordering and suffix separation has been done. We report a few early observations.

6.1 Reordering Errors

We have used reordering system developed by Patel et al. [25]. Table 8 details an example of the reordering error. In the example, the phrase sequence "very useful for losing fat" is wrongly reordered and that has resulted in a wrong translation. The wrong reordering not only affects the structure of the output but also badly affects the phrase translation.

6.2 Bad Split

The suffix separation system developed by Pimpale et al. [28] is used. For Tamil, it has a limited list of manually created suffixes, and hence it does not work for many words. As suffixes are crudely chopped without consideration of the validity of the remaining part, errors get introduced. Most of the errors belong to the category where the words get split because they end with a suffix from our list, though these were not meant to be processed. This causes sparsity of these genuine terms in the data and leads to a wrong translation of those. For example, a genuine word, say "*abcd*" is getting split into "*ab*" + "*cd*," which is a wrong split, as "*abcd*" is a proper noun and hence should not have been split. To avoid suffix separation of such words, the NNP POS tag was tried, but it stopped many other valid candidates from preprocessing. A word getting split at a wrong position was also one of the major error cases. For example, a word with character sequence "*pqrstu*" was getting split into "*pqr*" and "*stu*" instead of "*pqrs*" and "*tu*." In such cases, both suffixes "*stu*" and "*tu*" are valid, and so deciding on when it goes wrong is difficult.

7 Conclusion and Future Work

In this paper, we presented various systems for English to Hindi, Malayalam, Punjabi, and Tamil MT. Factored SMT with suffix separation and reordering performs better. Transliteration as postprocessing further helps improve the translation quality. Failure of factored SMT for English-Punjabi and English-Malayalam would be another thread of this work to be continued. Further, we plan to work toward improving the preprocessing and postprocessing techniques for better translation quality and extend the approach to other Indian languages.

Bibliography

[1] R. D. Brown, Corpus-driven splitting of compound words, in: *Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation*, pp. 12–21, 2002.

[2] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer and P. S. Roossin, A statistical approach to machine translation, *Comput. Linguist.* **16** (1990), 79–85.

[3] S. F. Chen and J. Goodman, An empirical study of smoothing techniques for language modeling, in: *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 310–318, 1996.

[4] M. Collins, P. Koehn and I. Kuerová, Clause restructuring for statistical machine translation, in: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 531–540, 2005.

[5] J. Dlougach and I. Galinskaya, Building a reordering system using tree-to-string hierarchical model, *arXiv preprint arXiv:1302.3057* (2013).

[6] N. Durrani, H. Sajjad, A. Fraser and H. Schmid, Hindi-to-Urdu machine translation through transliteration, in: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 465–474, 2010.

[7] N. Durrani, H. Sajjad, H. Hoang and P. Koehn, Integrating an unsupervised transliteration model into statistical machine translation, in: *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL 2014)*, 14, Association for Computational Linguistics, pp. 148–153, 2014.

[8] S. Green, D. Cer and C. D. Manning, Phrasal: a toolkit for new directions in statistical machine translation, in: *Proceedings of the Ninth Workshop on Statistical Machine Translation*, 2014.

[9] R. Gupta, R. N. Patel and R. Shah, Learning improved reordering models for Urdu, Farsi and Italian using SMT, in: *Proceedings of the First Workshop on Reordering for Statistical Machine Translation, COLING 2012*, 2012.

[10] K. Heafield, KenLM: faster and smaller language model queries, in: *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Association for Computational Linguistics, pp. 187–197, 2011.

[11] M. Khalilov and J. A. R. Fonollosa, Syntax-based reordering for statistical machine translation, *Comput. Speech Lang.* **25** (2011), 761–788.

[12] P. Koehn and K. Knight, Empirical methods for compound splitting, in: *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics – Volume 1*, Association for Computational Linguistics, pp. 187–193, 2003.

[13] P. Koehn and H. Hoang, Factored translation models, in: *EMNLP-CoNLL*, pp. 868–876, 2007.

[14] P. Koehn, F. J. Och and D. Marcu, Statistical phrase-based translation, in: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, Association for Computational Linguistics, pp. 48–54, 2003.

[15] M. A. Kumar, V. Dhanalakshmi, K. P. Soman and S. Rajendran, Factored statistical machine translation system for English to Tamil language, *Pertanika J. Soc. Sci. Hum.* **22** (2014), 1045–1061.

[16] M. A. Kumar, B. Premjith, S. Singh, S. Rajendran and K. P. Soman, An overview of the shared task on machine translation in Indian languages (MTIL) – 2017, *J. Intell. Syst.* **28** (2019), 455–464.

[17] A. Kunchukuttan, A. Mishra, R. Chatterjee, R. Shah and P. Bhattacharyya, Sata-anuvadak: tackling multiway translation of Indian languages, *Pan* **841** (2014), 4–135.

[18] G. Leusch, N. Ueffing and H. Ney, CDER: efficient MT evaluation using block movements, in: *EACL*, 2006. Available at: http://aclweb.org/anthology/E06-1031.

[19] D. Marcu and W. Wong, A phrase-based, joint probability model for statistical machine translation, in: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing – Volume 10*, Association for Computational Linguistics, pp. 133–139, 2002.

[20] G. Minnen, J. Carroll and D. Pearce, Applied morphological processing of English, *Nat. Lang. Eng.* **7** (2001), 207–223.

[21] P. Nakov and J. Tiedemann, 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: Short Papers – Volume 2*, Association for Computational Linguistics, Jeju Island, South Korea, pp. 301–305, 2012.

[22] S. Nießen and H. Ney, Statistical machine translation with scarce resources using morpho-syntactic information, *Comput. Linguist.* **30** (2004), 181–204.

[23] F. J. Och, Minimum error rate training in statistical machine translation, in: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics – Volume 1*, Association for Computational Linguistics, pp. 160–167, 2003.

[24] K. Papineni, S. Roukos, T. Ward and W.- J. Zhu, BLEU: a method for automatic evaluation of machine translation, in: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 311–318, 2002.

[25] R. N. Patel, R. Gupta, P. B. Pimpale and M. Sasikumar, Reordering rules for English-Hindi SMT, in: *Proceedings of the Second Workshop on Hybrid Approaches to Translation*, Association for Computational Linguistics, pp. 34–41, 2013.

[26] R. N. Patel, P. B. Pimpale and M. Sasikumar, Statistical machine translation for Indian languages: mission Hindi, in: *Proceedings of NLP Tools Contest, ICON-2014: 11th International Conference on Natural Language Processing*, Goa University, India, 2014.

[27] R. N. Patel, P. B. Pimpale and M. Sasikumar, Statistical machine translation for Indian languages: mission Hindi, *arXiv preprint arXiv:1610.07418*, 2016.

[28] P. B. Pimpale, R. N. Patel and M. Sasikumar, SMT from agglutinative languages: use of suffix separation and word splitting, in: *Proceedings of ICON-2014: 11th International Conference on Natural Language Processing*, Goa University, India, 2014.
[29] M. Popovic and H. Ney, Towards the use of word stems and suffixes for statistical machine translation, in: *LREC*, Lisbon, Portugal, 2004.
[30] M. Popovic and H. Ney, POS-based word reorderings for statistical machine translation, in: *International Conference on Language Resources and Evaluation*, pp. 1278–1283, 2006.
[31] M. Popovic and H. Ney, Statistical machine translation with a small amount of bilingual training data, in: *5th LREC SALTMIL Workshop on Minority Languages*, pp. 25–29, 2006.
[32] M. Popovic and H. Ney, Word error rates: decomposition over POS classes and applications for error analysis, in: *Proceedings of the Second Workshop on Statistical Machine Translation*, Association for Computational Linguistics, pp. 48–55, 2007.
[33] M. Popovic, D. Stein and H. Ney, Statistical machine translation of German compound words, *Lect. Notes Computer Sci.* **4139** (2006), 616–624.
[34] M. Post, Y. Cao and G. Kumar, Joshua 6: a phrase-based and hierarchical statistical machine translation system, *Prague Bull. Math. Linguist.* **104** (2015), 5–16.
[35] B. Premjith, S. S. Kumar, R. Shyam, M. A. Kumar and K. P. Soman, A fast and efficient framework for creating parallel corpus, *Indian J. Sci. Technol.* **9** (2016), 1–7.
[36] T. Rama, K. Gali and P. V. S. Avinesh, Does syntactic knowledge help English-Hindi SMT?, *arXiv preprint arXiv:1401.4869* (2014).
[37] A. Ramanathan and D. D. Rao, A lightweight stemmer for Hindi, in: *The Proceedings of EACL*, 2003.
[38] A. Ramanathan, J. Hegde, R. M. Shah, P. Bhattacharyya and M. Sasikumar, Simple syntactic and morphological processing can help English-Hindi statistical machine translation, in: *Proceedings of International Joint Conference on NLP (IJCNLP08)*, pp. 513–520, 2008.
[39] M. Snover, B. Dorr, R. Schwartz, L. Micciulla and J. Makhoul, A study of translation edit rate with targeted human annotation, in: *Proceedings of Association for Machine Translation in the Americas*, pp. 200, 2006.
[40] C. Wang, M. Collins and P. Koehn, Chinese syntactic reordering for statistical machine translation, in: *EMNLP-CoNLL*, pp. 737–745, 2007.
[41] F. Xia and M. McCord, Improving a statistical MT system with automatically learned rewrite patterns, in: *Proceedings of the 20th International Conference on Computational Linguistics*, Association for Computational Linguistics, p. 508, 2004.