SMT vs NMT: A Comparison over Hindi & Bengali Simple Sentences

Sainik Kumar Mahata[1], Soumil Mandal[2], Dipankar Das[3], Sivaji Bandyopadhyay[4]
[134]Jadavpur University, Kolkata
[2]SRM University, Chennai
[1]sainik.mahata@gmail.com, [2]soumil.mandal@gmail.com,
[3]dipankar.dipnil2005@gmail.com, [4]sivaji_cse_ju@yahoo.com

Abstract

In this article, we identified the qualitative differences between Statistical Machine Translation (SMT) and Neural Machine Translation (NMT) outputs. We have tried to answer two important questions: **1.** Does NMT perform equivalently well with respect to SMT and **2.** Does using simple sentences as training units, add extra flavor in improving the quality of Machine Translation output. In order to obtain insights, we have developed three core models viz., SMT model based on Moses toolkit, followed by character and word level NMT models. All of the systems use English-Hindi and English-Bengali language pairs containing simple sentences as well as sentences of other complexity. In order to preserve the translations semantics with respect to the target words of a sentence, we have employed soft-attention into our word level NMT model. We have further evaluated all the systems with respect to the scenarios where they succeed and fail. Finally, the quality of translation has been validated using BLEU and TER metrics along with manual parameters like fluency, adequacy etc. We observed that NMT outperforms SMT in case of simple sentences whereas SMT outperforms in case of all sentence types.

1 Introduction

Machine Translation (MT) refers to automated translation. It is the process by which computer software is used to translate a text from one natural language (such as English) into another (such as Spanish). Translation itself is a challenging task for humans, and hence, is more challenging for computers. High quality translation requires a thorough understanding of syntax and semantical properties of both the source and target languages.

The importance of studying and developing better MT systems has gained popularity in the recent past due to rapid globalization, where people from multiple backgrounds having varying language knowledge are working together. Primarily two paradigms are currently followed for building MT systems. One is based on statistical techniques, while the other employs artificial neural networks.

The statistical model, commonly referred to as Statistical Machine Translation (SMT) (Weaver, 1955), addresses this challenge by creating statistical models, whose input parameters are derived from the analysis of parallel bilingual text corpora (Mahata et al., 2017). Some of the notable works on SMT are (Al-Onaizan et al., 1999; Lopez, 2008; Koehn, 2009), where the authors have dived deep into the challenges, working principles and possible improvements. SMT has shown good results for many language pairs and is responsible for the recent surge in the popularity of MT among general public .

On the other hand, despite being relatively new, Neural Machine Translation (NMT) (Bahdanau et al., 2014) has already shown promising results (Mahata et al., 2016; Wu et al., 2016) and hence has gained substantial attention and interest. Continuous recurrent models for translation, which do not depend on alignment or phrasal translation units, was introduced by Kalchbrenner and Blunsom (2013). The problem of rare word occurrence was addressed by Luong et al. (2014) and the effectiveness of global and local approach was explored by Luong et al. (2015). He et al. (2016) demonstrated a log-linear framework incorporating SMT features combined with NMT which ad-

175

dresses out of vocabulary and inadequate translation. The properties of these architecture was discussed in detail in (Cho et al., 2014). This approach generally creates much more accurate translation than SMT given sufficient amount of training data (Vaswani et al., 2013; Liu et al., 2014; Doherty et al., 2010).

In the current work, we have tested the performance of SMT and NMT on simple sentences (see Section 2) extracted from English-Hindi (En-Hn) and English-Bengali (En-Bn) parallel corpus provided by TDIL[1]. These experiments were done to dive into the scenarios where NMT and SMT outperform each other. Moreover, they would also help us in evaluating the question that whether usage of simple sentences, when training the MT models, creates any difference in the quality of the MT output.

We have constrained our language domain to Hindi and Bengali as these languages are used primarily in the Indian sub-continent. Number of native speakers of Hindi in India is 41.1% while that of Bengali is 8.11%. Hindi is written in Devanagari[2] script and Bengali is written in Eastern Nagari[3] script.

To test the effectiveness of the case study, SMT and NMT systems were also trained for the whole corpus, which consists of sentences with mixed complexity. For both simple sentence corpus and the whole corpus, BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and manual evaluation metrics like fluency and adequecy were calculated to validate the observed results.

The paper has been organized as follows. Section 2 describes the extraction of simple sentences from the parallel corpus given by TDIL. Section 3 and Section 4, describes the methodology for the training of the SMT and the NMT models respectively. Later, Section 5 and Section 6 describes the evaluation and conclusion, respectively.

2 Extraction of Simple Sentence Pairs

Since we wanted to analyze and compare both the models, SMT and NMT, with respect to how they perform on simple sentences, we first needed to extract such instances from our dataset that had data of varying complexity.

A simple sentence in this context is defined as a sentence which contains only one independent clause and has no dependent clauses. Generally, whenever two or more clauses are joined by conjunctions (coordinating and subordinating), it becomes a complex or a compound sentence accordingly. So, to get a hold on handling the conjunctions, we used the Stanford Dependency Parser [4] library to chunk the English sentences into phrases. (viz. NP (Noun Phrase), VP (Verb Phrase), PP (Preposition Phrase), ADJP (Adjective Phrase) and ADVP (Adverb Phrase)).
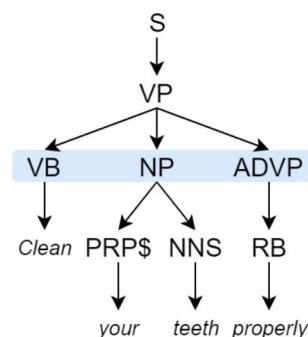

Figure 1: Extraction of phrase chunks.

We noticed that, simple sentences have an unique phrase structure that consists of combinations of NP, VP and PP. In conjunction with this theory, we applied two methods (viz. rule based approach and deep learning based approach) to extract simple sentences from the English corpus. The approaches are discussed in Section 2.1 and Section 2.2, respectively.

2.1 Rule Based Approach

We subjected 3046 simple sentences, extracted from various websites, to chunking using Stanford Dependency Parser (Manning et al., 2014), and extracted the unique phrase structures, which became the rules by which we further mined for simple sentences from the English corpus.

We extracted 205 unique rules, the surface forms of which, along with its Confidence Score, are shown in Table 1. The confidence score of the rules were calculated using

$$ConfidenceScore = \frac{No.OfSentencesPertainingToARule}{TotalNo.OfSentencesInTheTestData}$$

We tested our system on 2876 sentences (1438 simple sentences and 1438 complex/compound sentences) and got an accuracy of 89.22%. Table

2 shows the various validation metrics. Using this system, 10,349 simple sentences from the TDIL English corpus was extracted, as shown in Table 4.

Rules	Confidence
PP NP* PP VP NP*	8.40
PP NP* VP PP NP*	9.49
ADVP NP* VP* ADVP NP*	9.36
NP VP PP NP PP NP	12.15
NP ADVP VP* NP*	11.69
NP* VP NP*	11.69
NP* PP NP VP* NP	11.46
NP VP PP NP*	11.23
VP* NP* PRP* ADVP*	4.92
NP VP* NP* PP* ADJP* ADVP*	9.62

Table 1: Surface forms of the extracted rules. "*" means one or more occurrence of item.

	Other	Simple	Prec.	Kappa
Other	1275	90	93.41%	0.78
Simple	220	1291		
Recall	85.28%			
Acc.	89.22%			
F1	89.16%			

Table 2: Confusion matrix for the rule based approach.

2.2 Deep Learning Based Approach

We preferred Deep Learning approach over traditional Machine Learning (ML) approach as because in the ML approach we could only extract syntactic features, which was already exploited in the rule based approach discussed in Section 2.1. On the other hand, a deep learning technique learn categories incrementally through its hidden layer architecture. We wanted the deep learning framework to learn from the POS tags itself as it automatically clusters similar data into separate spaces.

For the deep learning model, we trained a multilayer feed-forward neural network with stochastic gradient descent (Bottou, 2010) as optimizer with back-propagation. The network contained two hidden layers of sizes 50 and 50 respectively. The activation function used was *tanh* and loss function used was *Mean Squared Error*. *Learning Rate* was kept at 0.001 and *number of epochs* were fixed at 100. The *batch size* was kept

at 128. The training data consisted phrases of 2876 sentences (1438 simple sentences and 1438 other complex/compound sentences). The trained model was subjected to 10 fold cross validation and it yielded an accuracy figure of 92.11%. Table 3 shows the other important validation metric measures.

	Other	Simple	Prec.	Kappa
Other	1287	76	92.22%	0.84
Simple	151	1362		
Recall	92.11%			
Acc.	92.11%			
F1	92.16%			

Table 3: Confusion matrix for deep learning based approach.

The TDIL English corpus was fed to this model and it yielded 14,976 simple sentences as shown in Table 4.

# of sentences		49999
# of other sentences	RL	39650
# of simple sentences	RL	10349
# of other sentences	DL	35023
# of simple sentences	DL	14976

Table 4: Simple Sentence Count

The deep learning based approach was preferred as it resulted in better accuracy. The Bengali and Hindi counterparts of these sentences were extracted to build a parallel corpus comprising of simple sentences only. The next step was to build MT models using this data, as well as the data from the whole corpus, and compare their respective results.

3 Statistical Machine Translation

Moses (Koehn et al., 2007) is a statistical machine translation system that allows us to automatically train translation models for any language pair, making use of a large collection of translated texts (parallel corpus). Once the model has been trained, an efficient beam search algorithm quickly finds the highest probability translation among the exponential number of choices.

For training the SMT model, we used English as the source language and Bengali and Hindi as the target languages. To prepare the data for training the SMT system, we performed the following steps.

3.1 Preprocessing

The following steps were employed to preprocess the Source and the Target texts.

- **Tokenization**: Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens. In our case, these tokens were words, punctuation marks, numbers.

- **Truecasing**: This refers to the process of restoring case information to badly-cased or non-cased text (Lita et al., 2003). Truecasing helps in reducing data sparsity.

- **Cleaning**: Long sentences (# of tokens > 80) were removed.

3.2 Language Model

Post these steps a *Language Model* (LM) was built using the target language, Bengali and Hindi, in our case, to ensure fluent output. KenLM (Heafield, 2011), which comes bundled with the Moses toolkit, was used for building this model.

3.3 Word Alignment and Phrase Table Generation

For word alignment in the translation model, GIZA++ (Och and Ney, 2003) was used. Finally, the phrase table was created and probability scores were calculated. Training the Moses statistical MT system resulted in the generation of two models, one is a Phrase Model and the other is a Translation Model. Moses scores the phrase in the phrase table with respect to a given source sentence and produces best scored phrases as output.

The results of this system when trained and tested on the simple sentence corpus and the general corpus, for both En-Bn and En-Hn language pairs. The results and evaluation of the systems are shown in Sec 5, Table 5 and Table 6.

4 Neural Machine Translation

Neural machine translation (NMT) is an approach to machine translation that uses neural networks to predict the likelihood of a sequence of words, typically modeling entire sentences in a single integrated model. NMT departs from traditional phrase-based statistical approaches in that uses separately engineered subcomponents like Language Model generation, Word Alignment and Phrase Table generation. The main functionality of NMT is based on the sequence to sequence

(seq2seq) architecture, which is described in Section 4.1.

4.1 Seq2Seq Model

The sequence to sequence model is a relatively new idea for sequence learning using neural networks. It has gained quite some popularity since it achieved state of the art results in machine translation task. Essentially, the model takes as input a sequence

$$X = \{x_1, x_2, ..., x_n\}$$

and tries to generate the target sequence as output

$$Y = \{y_1, y_2, ..., y_m\}$$

where x_i and y_i are the input and target symbols respectively. The architecture of seq2seq model comprises of two parts, the encoder and decoder. We experimented with two types of NMT models (word and character level), with both using the seq2seq architecture, the difference being in the inputs to its encoder and decoder. They are discussed in the sections 3 and 4 below. The working of seq2seq architecture at the word level is shown in Fig. 2. We implemented both the models using the Keras (Chollet et al., 2015) library.

4.1.1 Word Level NMT

To build our world level NMT model, we used the seq2seq with attention mechanism. This architecture has recently shown to achieve state of the art quality translation across many different language pairs. The details of the seq2seq model along with the training details are given below.

Encoder The encoder takes a variable length sequence as input and encodes it into a fixed length vector, which is supposed to summarize it's meaning and taking into account it's context as well. A Long Short Term Memory (LSTM) cell was used to achieve this. The directional encoder reads the sequence from one end to the other (left to right in our case),

$$\vec{h}_t = \vec{f}_{enc}(E_x(x_t), \vec{h}_{t-1})$$

Here, E_x is the input embedding lookup table (dictionary), $\vec{f}_{enc}$ are the transfer function for the Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) recurrent unit. A contiguous sequence of encodings C is constructed which is then passed on to the decoder.
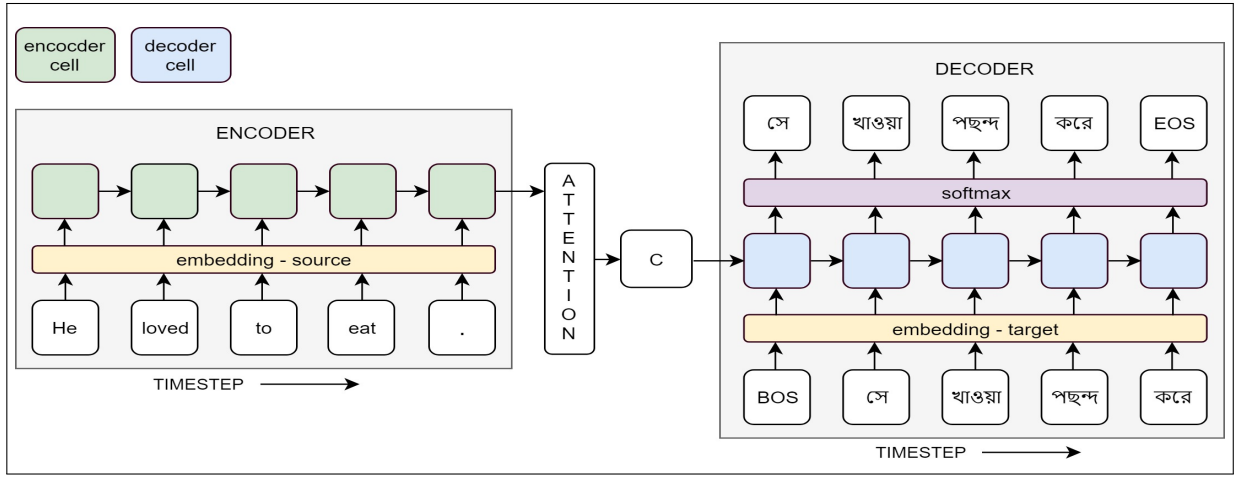
Figure 2: NMT with attention architecture.

Decoder The decoder takes as input, the context vector C from the encoder, and computes the hidden state at time t as,

$$s_t = f_{dec}(E_y(y_{t-1}), s_{t-1}, c_t)$$

Subsequently, a parametric function out_k returns the conditional probability using the next target symbol k.

$$(y_t = k \mid y < t, X) = \frac{1}{Z} exp(out_k(E_y(y_t - 1), s_t, c_t))$$

Z is the normalizing constant,

$$\sum_j exp(out_j(E_y(y_t - 1), s_t, c_t))$$

The entire model can be trained end-to-end by minimizing the log likelihood which is defined as

$$L = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{Ty^n} log p(y_t = y_t^n, y_{jt}^n, X^n)$$

where N is the number of sentence pairs, and X^n and y_t^n are the input sentence and the t-th target symbol in the n-th pair respectively.

Training For training our model, we used the seq2seq with attention architecture with LSTM cell. We used two LSTM cells, stacked upon each other, where one acts as the encoder and the other as the decoder. We trained our model on 14976 data (for simple sentence corpus), 49999 sentences (for Bengali and Hindi general corpus), *batch size* at 256, *number of epochs* at 100 and *learning rate* at 0.001. The activation function used was *softmax*, optimizer used was *rmsprop* and the loss calculation at each step was done using *categorical cross-entropy*.

Attention Neural processes involving attention (Vaswani et al., 2017) has been largely studied in computational neuro-science. This concept is very loosely based on visual attention mechanism in humans. With attention mechanism, the need to encode the full source sentence into a fixed length vector is omitted. Rather we allow the decoder to attend different parts of the source sentence at each time step of the output generation. Essentially, we let the model learn what to *attent* to based on the input sequence and what is predicted so far.

Mathematically, it computes the context vector c_t at each time step t as a weighted sum of the source hidden states,

$$c_t = \sum_{t=1}^{Tx} \alpha_t h_t$$

Each attention weight α_t represents how much relevant the t-th source token x_t is to the t-th target token y_t and is computed as :

$$\alpha_t = \frac{1}{Z} exp(score(E_y(y_t - 1), s_{t-1}, h_t))$$

where

$$Z = \sum_{k=1}^{Tx} exp(score(E_y(y_t - 1), s_{t-1}, h_k))$$

Z is the normalization constant. score() is a feed forward neural network with a single hidden layer that scores how well the source symbol x_x and the target symbol y_t match. E_y is the target embedding lookup table and s_t is the target hidden state at time t.

The results and evaluation of the systems are shown in Sec 5.

179

4.1.2 Character Level NMT

Character level NMT (CNMT) performs better than Word Level NMT due to the following reasons

1. It does not suffer from out-of-vocabulary issues

2. It is able to model different, rare morphological variants of a word

3. It does not require segmentation (Chung et al., 2016).

Generally, CNMT works best when majority of alphabets, in the source and target language, overlap i.e both the languages share a common or similar script. Still, we tried to find out its performance on the simple sentence and whole corpus, though in our case Nagari script and Roman script utilizes completely different alphabets. The model has two parts (encoder and decoder) as discussed below.

Encoder For building the encoder we used LSTM cells. The input of the cell was one hot tensor of English sentences (embeddings at character level). From the encoder, the internal states of each cell were preserved and the outputs were discarded. The purpose of this is to preserve the information at context level. These states were then passed on to the decoder cell as initial states.

Decoder For building the decoder, again an LSTM cell was used with initial states as the hidden states from encoder. It was designed to return both sequences and states. The input to the decoder was one hot tensor (embeddings at character level) of Bengali and Hindi sentences while the target data was identical, but with an offset of one time-step ahead. The information for generation is gathered from the initial states passed on by the encoder. Thus, the decoder learns to generate target data [t+1,...] given targets [..., t] conditioned on the input sequence. It essentially predicts the output sequence, one character per output time step.

Training For training the model, *batch size* was set to 64, *number of epochs* was set to 100, activation function was *softmax*, optimizer chosen was *rmsprop* and loss function used was *categorical cross-entropy*. Learning rate was set to 0.001.

The results and evaluation of the systems are shown in Sec 5.

5 Evaluation and Analysis

All of our translation systems were evaluated in two ways, automatic and manual, depiction s of which are discussed in the section below.

5.1 Automatic Evaluation

Automatic evaluation was done by scoring the translations using BLEU and TER metrics. The results are shown in Table 5 and 6. In the tables, "Bn" and "Hn" means Bengali and Hindi respectively. "CNMT" and "WNMT" means character and word level NMT models respectively. Presence of Attention mechanism in the model is denoted using "A" and the contrary is denoted using "NA"

Model	Simple Sent.		Whole Corp.	
(Bn)	BLEU	TER	BLEU	TER
SMT	0	117.67	15.9	85.26
CNMT (NA)	8.69	91.87	4.19	88.22
WNMT (NA)	9.68	86.84	3.61	98.03
WNMT (A)	9.95	85.66	3.77	96.72

Table 5: Automatic evaluation metrics for En-Bn Model.

Model	Simple Sent.		Whole Corp.	
(Hn)	BLEU	TER	BLEU	TER
SMT	3.98	101.945	12.86	95.092
CNMT (NA)	7.98	92.85	5.96	85.18
WNMT (NA)	10.01	90.28	4.87	96.97
WNMT (A)	10.54	90.26	5.21	94.20

Table 6: Automatic evaluation metrics for En-Hn Model.

5.2 Manual Evaluation

Translation quality was judged by four linguists. Two had Bengali mother tongue (evaluated Bn model), while the other two had Hindi mother tongue (evaluated Hn model). The evaluation criterion were Adequacy and Fluency. Adequacy means how much of the meaning expressed in the target translation. Fluency means to what extent the translation is well-formed grammatically, contains correct spellings and intuitively acceptable and can be sensibly interpreted by a native speaker. The speakers were asked to rate the translation in range of 1-5, where '1' is the lowest and '5' is the highest. The manual evaluation measures for English-Bengali and English-Hindi language pair

Model (Bn)	SMT		CNMT		WNMT (NA)		WNMT(A)	
Corpus	Simple	Whole	Simple	Whole	Simple	Whole	Simple	Whole
Adequecy 1	0	2.15	1.98	1.54	2.02	1.44	2.15	1.47
Fluency 1	0	1.87	2.27	1.98	2.36	1.86	1.98	2.02
Adequecy 2	0	2.24	1.87	1.66	1.96	1.57	2.01	1.69
Fluency 2	0	1.92	2.05	1.86	2.21	1.77	2.26	1.93
Avg. Adequecy	0	2.195	1.925	1.6	1.99	1.505	2.08	1.58
Avg. Fluency	0	1.895	2.16	1.92	2.285	1.815	2.12	1.975

Table 7: Depiction of Manual Evaluation conducted by Bengali language speaking experts.

Model (Hn)	SMT		CNMT		WNMT(NA)		WNMT(A)	
Corpus	Simple	Whole	Simple	Whole	Simple	Whole	Simple	Whole
Adequecy 1	0.8	2.06	1.96	1.69	2.36	1.47	2.26	1.49
Fluency 1	0.5	1.72	2.04	2.08	2.27	1.92	2	2.22
Adequecy 2	1.02	2.18	1.79	1.71	2.02	1.63	2.18	1.9
Fluency 2	0.65	1.98	2.1	1.94	2.39	1.83	2.33	1.87
Avg. Adequecy	0.91	2.12	1.875	1.7	2.19	1.55	2.22	1.695
Avg. Fluency	0.575	1.85	2.07	2.01	2.33	1.875	2.165	2.045

Table 8: Depiction of Manual Evaluation conducted by Hindi language speaking experts.

is given in Table 7 and Table 8 respectively. In the tables, presence of Attention mechanism in the model is denoted using "A" and the contrary is denoted using "NA".

5.3 Analysis

We can clearly see in the results, that a NMT model, when trained using simple sentences, performs better than a SMT model, when trained using the same sentence pairs.

But, at the same time, SMT outperforms NMT, when trained using the whole corpus. This is due to the fact that NMT doesn't quite work well with less amount of data and highly complex sentences.

Similarly, we also see that character based NMT works better than word based NMT, when dealing with less amount of data. But again, we have to keep in mind that for a character based NMT to work well, we have to train it using a Source-Target language pair, who share a common script.

Further, word based NMT with attention perform relatively better than a character based NMT. We didn't use attention in the Character NMT, as attention won't be able to attent to individual characters.

6 Conclusion and Future Work

In this work, we have tried to analyze the scenarios where SMT performs better than NMT and vice-versa. Also, we have tried to find out whether MT

models give better outputs when trained with simple sentences rather than when trained using sentences of various complexities.

As a future prospect, we would like to take the "other" (Complex+Compound) sentence pairs and simplify it, so that the whole MT models can be trained using more simple sentences. Also, we would like to increase the number of LSTM encoding and decoding layers as well as include embeddings like ConceptNet[5] in our future works.

Acknowledgments

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A Smith, and David Yarowsky. 1999. Statistical machine translation. In *Final Report, JHU Summer Workshop*, volume 30.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

[5]https://github.com/commonsense/conceptnet-numberbatch

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

François Chollet et al. 2015. Keras. https://keras.io.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. *CoRR*, abs/1603.06147.

Stephen Doherty, Sharon O?Brien, and Michael Carl. 2010. Eye tracking as an mt evaluation technique. *Machine translation*, 24(1):1–13.

Wei He, Zhongjun He, Hua Wu, and Haifeng Wang. 2016. Improved neural machine translation with smt features. In *AAAI*, pages 151–157.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*, volume 3, page 413.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Lucian Vlad Lita, Abe Ittycheriah, Salim Roukos, and Nanda Kambhatla. 2003. Truecasing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 152–159. Association for Computational Linguistics.

Shujie Liu, Nan Yang, Mu Li, and Ming Zhou. 2014. A recursive recurrent neural network for statistical machine translation.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):8.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

Sainik Mahata, Dipankar Das, and Santanu Pal. 2016. Wmt2016: A hybrid approach to bilingual document alignment. In *WMT*, pages 724–727.

Sainik Kumar Mahata, Dipankar Das, and Sivaji Bandyopadhyay. 2017. Bucc2017: A hybrid approach for identifying parallel sentences in comparable corpora. *ACL 2017*, page 56.

Sainik Kumar Mahata, Dipankar Das, and Sivaji Bandyopadhyay. 2018. Mtil2017: Machine translation using recurrent neural network on statistical machine translation. *Journal of Intelligent Systems*, pages 1–7.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. pages 311–318.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with large-scale neural language models improves translation. In *EMNLP*, pages 1387–1392.

Warren Weaver. 1955. Translation. *Machine translation of languages*, 14:15–23.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.