Domain Adaptation of NMT models for English-Hindi Machine Translation Task at AdapMT ICON 2020

Ramchandra Joshi[1], **Rushabh Karnavat**[1], **Kaustubh Jirapure**[1], **Raviraj Joshi**[2]
[1]Pune Institute of Computer Technology, Pune
[2]Indian Institute of Technology Madras, Chennai
{rbjoshi1309,rpkarnavat,kaustubhmjirapure,ravirajoshi}@gmail.com

Abstract

Recent advancements in Neural Machine Translation (NMT) models have proved to produce a state of the art results on machine translation for low resource Indian languages. This paper describes the neural machine translation systems for the English-Hindi language presented in AdapMT Shared Task ICON 2020. The shared task aims to build a translation system for Indian languages in specific domains like Artificial Intelligence (AI) and Chemistry using a small in-domain parallel corpus. We evaluated the effectiveness of two popular NMT models i.e, LSTM, and Transformer architectures for the English-Hindi machine translation task based on BLEU scores. We train these models primarily using the out of domain data and employ simple domain adaptation techniques based on the characteristics of the in-domain dataset. The fine-tuning and mixed-domain data approaches are used for domain adaptation. Our team was ranked first in the chemistry and general domain En-Hi translation task and second in the AI domain En-Hi translation task.

1 Introduction

Machine understanding of natural language queries is of paramount importance to automate different workflows. The natural language query can be in the form of text or speech. Processing of query in the form of text is more popular and easy than directly processing the raw speech waveform. The text-based Natural Language Processing (NLP) include tasks like classification, token tagging, summarization, and translation. Machine translation is an NLP technique to translate a sentence from a source language to a target language. The Neural Machine Translation (NMT) is a recent approach to translation producing state of the art results (Bahdanau et al., 2014). NMT defines translation as a sequence to sequence task and uses sequence-based neural architectures like Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017). Traditional techniques like rule-based translation and Statistical Machine Translation (SMT) have been outperformed by NMT models achieving significant improvements on MT tasks. In this work, we are specifically concerned with English-Hindi neural translation.

The Hindi language is one of the most popular languages in India and the fourth most spoken language in the world. Hindi is native to India and is spoken by more than 550 million total speakers worldwide. However, the number is much less as compared to global languages like English. On similar lines, the training data for the Hindi language that is publicly available for MT tasks is relatively less as compared to other highly popular languages worldwide like English, French, and German. This is important as MT tasks require a huge amount of training data to produce remarkable results using NMT models. Hindi being a relatively low resource and morphologically rich language, the amount of research in MT tasks for the Hindi language is limited (Philip et al., 2019). As Hindi is the most widely spoken language in the Indian subcontinent and the majority of content across the globe is published in English, the research in MT tasks for English-Hindi language pair becomes highly important.

Domain adaptation of translation systems to specific domains is a common practice for low resource language pairs. The adaptation is relevant as the text in different domains can vary widely (Luong and Manning, 2015). For example, social media text and the text in literary work will be quite different from style, grammar, and abbreviations perspective. The domains can be distinguished based on topics like politics, life science, news, etc, or the style of writing like formal and informal. A

Proceedings of the 17th International Conference on Natural Language Processing: Adap-MT 2020 Shared Task, pages 11–16
Patna, India, December 18 - 21, 2020. ©2020 NLP Association of India (NLPAI)

translation model trained on one domain may not work well on other domains. The problem is more severe in models that use word-based representation as most of the domain-specific words will be out of vocabulary (Sennrich et al., 2015b). In this work, we explore ideas for domain adaptation for English-Hindi translation on the AdapMT Shared Task ICON 2020 data sets.

The AdapMT Shared Task ICON 2020 aims to evaluate the capability of general domain machine translation for Low Resource Indian Languages. Indian languages considered in AdapMT Shared Task ICON 2020 for translation are English-Hindi, English-Telugu, and Hindi-Telugu. The shared task also focuses on Low Resource domain adaptation of machine translation systems. The adaptation is done with the use of already publicly available parallel corpora and some small in-domain parallel data for AI and Chemistry domains. The creation of a publicly available parallel corpus for low resource Indian languages is another important goal of this task.

This paper describes the system built for the English-Hindi general MT and domain adaptation tasks held under AdapMT Shared Task ICON 2020. We experimented with two popular NMT models namely attention-based LSTM encoder-decoder architecture and the Transformer architecture. For domain adaption, we explore fine-tuning and mixed domain training approaches. We show that the mixed domain training performs better than the fine-tuning based approach for the datasets used in this work.

2 Architecture

In this section, we describe the two popular seq2seq neural architectures for machine translation used in this work. The encoder-decoder architecture consisting of a source side encoder and a target side decoder is used for the sequence to sequence tasks (Sutskever et al., 2014). The encoder encodes the text in a source language into a latent representation which is consumed by the decoder to generate the text in the target language. The decoder acts like a contextual language model generating target text by attending to the source representations. The attention mechanism is thus an integral part of encoder-decoder models which allows the decoder to focus on the right context while generating the corresponding target token.

2.1 LSTM model

The LSTM based encoder-decoder models use stacked LSTM layers on both encoder and decoder sides. The LSTM and GRU are commonly used recurrent neural network architectures for machine translation. In this work, we use LSTM based recurrent architecture as it is shown to give slightly better results (Britz et al., 2017). The series of stacked LSTM layers encode the source text. The hidden state of the last LSTM layer is used as the encoded output. Subsequently, the target sequence is decoded sequentially using stacked LSTM layers. The decoder also makes use of an attention mechanism to attend to the encoder's hidden state. The additive attention and dot product attention are widely used attention mechanisms (Bahdanau et al., 2014; Luong et al., 2015). In this work, we restrict ourselves to the use of additive attention.

2.2 Transformer model

The recently introduced Transformer model has found a home in almost all NLP tasks starting with neural machine translation (Vaswani et al., 2017). It has helped advance the state of the art in NLP and even employed for speech and vision tasks (Karita et al., 2019; Ramachandran et al., 2019). The Transformer uses the self-attention mechanism as the single most important component. For the task of translation, the Transformer is used on both the encoder and decoder side. It comprises various encoders and decoders stacked over each other. The main advantage of Transformer over LSTM is the parallelism on the encoder side which helps us fully exploit the underlying hardware. The multi-headed self-attention is another architectural change that helps in providing superior results as compared to LSTM.

On the encoder side, the input words are converted to vector embeddings and positional encoding is added to those embeddings so that the transformer gets the sense of the order or position of words. These embeddings are then passed on to the first encoder layer of the Transformer. The encoder consists of multi-head self-attention and a feed-forward neural network. The output from one encoder layer is given as input to the next encoder layer. The output of the final encoder layer is sent to the decoder.

The decoder consists of masked-multi head attention, multi-head Attention, and a feed-forward neural network. The embeddings along with positional encodings are passed on to the first layer of the decoder. The masked multi-head Attention mechanism only pays attention to the previous words. Then, it is passed through the multi-head attention mechanism attending to the encoder state and a feed-forward neural network. The output of the decoder is passed to the linear and the softmax layer where the vector scores are turned into probabilities and the word with the highest probability is chosen as output.

3 Domain Adaptation

While generalization is always desirable the machine learning systems are often biased towards the domain of the training data. Each domain has a different distribution and different domain data are mixed while building general systems. In very basic terms, the vocabulary of the different domains is mostly different. Table 2 shows the percentage of in-domain tokens that are present in publicly available English-Hindi parallel training corpus. Almost 20-40% of the tokens are specific to the target domain and not present in the general corpus. Some terms are specific to and most frequently used in a particular domain. For some words, the meaning may be different across domains. For example, *"As I said that here we have one hidden layer, you can have multiple hidden layers also"* is a sentence from AI domain. Whereas, *"Triacylglycerol contains three fatty acids that are esterified to the glycerol backbone"* is from the Chemistry domain. These two sentences are very specific to their domain and rarely use in real-life conversations. To interpret them in the best way we need a domain expert or subject matter expert. Similarly to build a system that works best on a particular domain, we need to make use of domain-specific data. Now because we have the same underlying language rules irrespective of the domain we can make use of out of domain data to enhance our systems if in-domain data is less. This is exactly where domain adaptation comes into the picture.

Domain adaptation is a form of transfer learning where we adapt a general system for a specific domain. That is we tune the model to adapt to the distribution of the target domain. It has been widely studied in the context of machine translation (Chu and Wang, 2018). The adaptation techniques can either be data or model-centric. The data related approaches try to exploit the monolingual corpus of the target domain (Domhan and Hieber, 2017). A commonly used technique is to use back-translation to expand the parallel corpus of the in-domain data (Sennrich et al., 2015a). The model-based approaches also make use of monolingual corpus from the target domain to train a language model and then do a shallow or deep fusion (Gulcehre et al., 2015). There is another set of training based technique which also go into model-centric approaches. In these approaches, the model is first trained on large out of domain parallel corpus and then re-trained or fine-tuned on the small in-domain parallel corpus. There are different variations proposed in literature where the second fine-tuning is done on a mixed parallel corpus instead of only using the in-domain corpus (Chu et al., 2017). The concept of domain tag was introduced in (Sennrich et al., 2016). The model is passed the domain label along with each training sample so that it learns to distinguish between the domains. The under-represented domains are oversampled. In this work, we evaluate the domain data fine-tuning approach and mixed-data training approach. In the first approach, we train the model on general corpus followed by in-domain corpus. In the second approach, we mixed the in-domain corpus with the general data and do a single training. Since the amount of in-domain data is very less as compared to the overall general or mixed-domain data we oversample in-domain examples while training.

4 Experimental Setup

4.1 Dataset Details

In our English to Hindi machine translation experiments, we have used the publicly available IIT Bombay (IITB) English-Hindi Parallel Corpus (Kunchukuttan et al., 2017). The training data in the IITB corpus consists of nearly 1.5M training samples. The IITB training data consists of sentences from the various domain.

In addition to this, we have also used the AI and Chemistry in-domain parallel corpus provided by AdapMT Shared Task ICON 2020 organizers for training and testing the models for respective domains. The AI in-domain corpus contains 4872, 400, and 401 sentences in the train, validation, and test set, respectively. The Chemistry in-domain corpus contains 4984, 300, and 397 sentences in the train, validation, and test set, respectively. The data set details are described in Table 1.

Data	Sentences	~Tokens
IIT Bombay Train	1561840	19.85M / 21.4M
General Test	507	9k / -
AI Train	4872	77k / 83k
AI Dev	400	6k / 6k
AI Test	401	7k / -
Chemistry Train	4984	125k / 139k
Chemistry Dev	300	7k / 8k
Chemistry Test	397	7k / -

Table 1: Statistics of the Data (En / Hi)

Data	AI	Chemistry	General
Train (U)	47 / 68	64 / 60	-
Dev (U)	58 / 80	78 / 76	-
Test (U)	59 / -	55 / -	56 / -
Train	78 / 90	81 / 86	-
Dev	77 / 90	81 / 87	-
Test	78 / -	77 / -	76 / -

Table 2: Approx. % of AdapMT domain dataset tokens (En / Hi) present in IITB Train data. Rows with a suffix 'U' indicates unique tokens, while data with no suffix indicates all tokens

4.2 Data Processing

The individual data samples are lowercased followed by the removal of all the special characters. For training purposes, we exempted all the sentences from IITB English-Hindi Parallel Corpus with a length greater than 20 words. This was mainly done because of resource constraints to speed up training. After pre-processing, we train a sentence piece sub-word tokenizer to tokenize the English, as well as Hindi sentences citekudo2018sentencepiece. We train a unigram based tokenizer with a vocab size 32k (Kudo, 2018). The source and target corpus of the IITB parallel corpus was used to train the individual sentence piece models. For experiments involving domain adaptation, the domain data from the train set was also included in the sentence piece training data.

4.3 Training Details

In this paper, we used the LSTM and Transformers based models for the English to Hindi machine translation task. For the LSTM model-based experiments, we used an attention-based encoder-decoder LSTM architecture. The encoder side of LSTM is bi-directional and the decoder side of LSTM is unidirectional with Bahdanau additive attention mechanism. The number of layers on the encoder and decoder side is set to 1 with 512 hidden units in each layer. We have used a batch size of 128 and an embedding size of 256. Adam optimizer was used as an optimizer (Kingma and Ba, 2014). The subword tokenizer is used to get the subword tokens as it is known to handle the OOV problem well.

For the Transformer model, the encoder and decoder have 6 layers each and the number of hidden layers in each layer is set to 512. The batch size was set to 128. The number of heads used is 8 with a word embedding size of 512. The optimizer used was Adam. The models were implemented in Tensorflow 2.0 and trained for a maximum of 10 epochs. The validation loss was used to pick the best epoch. The standard greedy decoding was used for all the experiments. For longer sentences, during decoding, a simple heuristic to split the data at comma was used followed by separate translations. While this approach may not be well suited to the translation as the alignment is not always monotonous, it worked decently well given the nature of the in-domain sentences.

For our experiments with LSTM and Transformer models, we first trained the models on the IITB training corpus. The models are then retrained on in-domain AI and Chemistry parallel corpus to see the improvements in the machine translation model with the inclusion of small in-domain parallel data. In the second approach, the IITB corpus is mixed with the in-domain corpus individually and, a single training is performed. The in-domain corpus in oversampled 10 times to account for a very low in-domain corpus as compared to the general corpus.

5 Results and Discussion

We evaluate the mixed data and fine-tuning approaches on LSTM and Transformer NMT models. To compare the models Bilingual Evaluation Understudy (BLEU) score is used (Papineni et al., 2002). We report the BLEU score on validation data of AI and Chemistry in-domain corpus. Table 3 shows the results for de-tokenized validation data. The mixed data training approach performs the best in comparison to the no-domain data and fine-tuning approach. The no-domain data approach performs better as compared to the fine-tuning approach. This indicates that the simple fine-tuning approach is not suited to the very small in-domain corpus and is susceptible to catastrophic forgetting.

Model	AI dev	Che dev
LSTM (only IITB)	11.54	8.13
Transformer (only IITB)	10.66	4.73
LSTM (mixed)	**16.53**	**9.86**
Transformer (mixed)	12.68	5.07
LSTM (fine-tuning)	10.62	5.63
Transformer (fine-tuning)	11.60	4.88

Table 3: BLEU scores on in-domain dev data (model with the suffix 'only IITB' indicates that model is trained on samples from IITB train examples only, the model with the suffix 'mixed' indicates that the model is trained on data that is obtained by mixing oversampled in-domain training data with IITB training data, the model with suffix 'fine-tuning' indicates that the model is first trained on samples from IITB training data and then re-trained on in-domain corpus)

Data	General	AI	Chemistry
Test Data	14.81	19.08	13.95

Table 4: BLEU scores on test data as reported by AdapMT Shared Task ICON 2020 organizers

We see that although the Transformer based models perform well on the IITB test data they do not generalize well on the domain tasks. However, we feel that the low numbers with the Transformer can be enhanced using appropriate hyper-parameters and modifying the training approach. The system submitted for evaluation was LSTM based model trained on a mixed corpus which was giving the best validation scores. The results of the test system are shown in Table 4. The translations for the general test set were generated using the LSTM model trained only on the IITB parallel corpus.

6 Conclusion

In this paper, we evaluated the effectiveness of attention-based encoder-decoder LSTM and Transformer models on a low resource English to Hindi Translation Task held under AdapMT Shared Task ICON 2020. Our experiments showed that mixed domain training works well as compared to the fine-tuning approach for domain adaptation. The addition of small in-domain parallel data can indeed improve the results on AI and Chemistry domains provided in the shared task.

Acknowledgements

This work was done under the L3Cube Pune mentorship program. We would like to thank L3Cube and our mentors for the end to end guidance and encouragement to participate in the shared task.

References

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

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of simple domain adaptation methods for neural machine translation. *arXiv preprint arXiv:1701.03214*.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. *arXiv preprint arXiv:1806.00258*.

Tobias Domhan and Felix Hieber. 2017. Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1500–1505.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.

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

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. 2019. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 449–456. IEEE.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2017. The iit bombay english-hindi parallel corpus. *arXiv preprint arXiv:1710.02855*.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 76–79.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jerin Philip, Vinay P Namboodiri, and CV Jawahar. 2019. A baseline neural machine translation system for indian languages. *arXiv preprint arXiv:1907.12437*.

Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. 2019. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015a. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015b. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

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 5998–6008.