JUST System for WMT20 Chat Translation Task

Roweida Mohammed, Mahmoud Al-Ayyoub and Malak Abdullah
Jordan University of Science and Technology
Irbid, Jordan
roweida.221@gmail.com,{maalshbool, mabdullah}@just.edu.jo

Abstract

Machine Translation (MT) is a sub-field of Artificial Intelligence and Natural Language Processing that investigates and studies the ways of automatically translating a text from one language to another. In this paper, we present the details of our submission to the WMT20 Chat Translation Task, which consists of two language directions, English→German and German→English. The major feature of our system is applying a pre-trained BERT embedding with a bidirectional recurrent neural network. Our system ensembles three models, each with different hyperparameters. Despite being trained on a very small corpus, our model produces surprisingly good results.

1 Introduction

The language of chat texts is considered a common language where people are rarely paying attention to correct spelling. Therefore, using the traditional methods of Machine Translation (MT), like dictionaries, is insufficient (Hernández, 2009). As deep learning (DL) models are becoming more evolved and complex, this motivates the natural language processing (NLP) community researchers to employ them for challenging tasks such as MT of informal language, such as what is used in chat. Techniques like contextual word embeddings and pre-trained DL models are becoming very common in natural language generation (NLG) tasks such as MT (Kusner et al., 2015; Zou et al., 2013; Abdullah and Shaikh, 2018; Al-Bdour et al., 2019).

The Chat Translation Task is a new task in the Fifth Conference on Machine Translation (WMT20).[1] Translating chat text, specifically the chats of customer support, is a main and exciting task in the field of MT. This kind of tasks has not been widely considered in previous MT studies, mostly because of the absence of openly existing datasets. The target of this new Chat Translation Task is to translate the customer support chat text from English to German and vice versa. The essential goal of this task is to develop models that can translate conversational text and study the use of multilingual models.

We take part in the WMT20 shared chat translation task in two language directions: English→German and German→English. In this paper, we discuss our submission for this task, which is based on the bidirectional recurrent neural networks (bi-RNN) (Schuster and Paliwal, 1997) and using the pre-trained BERT embedding, known as bert-base-multilingual-cased (Devlin et al., 2018).

This paper is constructed as follows. In Section 2, the task and data descriptions are provided. Section 3 discusses our proposed model. Section 4 shows the experiments we conduct and their results. Finally, the Conclusion is in Section 5.

2 Task and Data Description

The Chat Translation shared task of WMT20 offers participants the opportunity to address a challenging problem faced by many companies today as they expand their customer support units to multiple different languages.

The shared task provides a dataset consisting of a set of conversations between agents and customers. The organizers supplied a corpus for the English-German language pair. Specifically, the task involves translating the chat text of an agent speaking English and a customer speaking German. We are asked to translate the agent's chat text from English to German, and the customer's from German to English.

The dataset used for this shared task depends on the corpus of Taskmaster-1 (Byrne et al., 2019),

[1] http://www.statmt.org/wmt20/chat-task.html

Proceedings of the 5th Conference on Machine Translation (WMT), pages 479–482
Online, November 19–20, 2020. ©2020 Association for Computational Linguistics

which has the English language, and it consists of dialogues in six fields. A small part of this dataset was chosen and translated to German. The shared task has been provided with train, development, and test sets in JSON format. Each chat in the data file has a specific structure. Table 1 shows the number of conversations in each file of the dataset.

Dataset	# of Conversation
Train dataset	550
Dev dataset	78
Test dataset	78

Table 1: Number of conversations in each set.

Each conversation contains a speaker (who is either an agent or a customer), a source chat text, and a target chat text. For the test set file, we are asked to translate the source chat text to target depending on the speaker. If it is an agent, the translation is from English to German. Otherwise, the translation is from German to English. For evaluating the participating models, the task organizers employ both automatic metrics (BLEU (Papineni et al., 2002) and TER (Snover et al., 2006)) as well as human evaluation.

3 JUST System

Our System follows the sequence of steps shown in Figure 1. In the following subsections, we discuss each step in details.

3.1 Preprocessing Data

For the dataset preprocessing, we first converted the files from JSON file, as given in the shared task, to text files, so we can work with them easily. The training, dev, and test sets are divided into two groups: one that contains the agent as the speaker (English→German) and one that contains the customer as the speaker (German→English). Table 2 shows the number of examples in each group.

Groups	Train	Dev	Test
Agent	7,629	1,040	1,133
Customer	6,215	862	967

Table 2: Number of examples in each group.

3.2 Extracting Features

After preparing the dataset and preprocessing it, we use the pre-trained BERT model to get the word embeddings of the dataset. Specifically, we use Bert-base-multilingual-cased[2] to extract feature vectors of the dataset to be used in the training of our models. For each word in the sentence of the encoder side, we get a file containing the word's embedding. The same is done for the decoder side.

3.3 The System Architecture

Our system is an adaptation of OpenNMT[3], an open-source toolkit for neural machine translation (NMT) (Klein et al., 2017). It is created on the PyTorch framework (Paszke et al., 2017). After ensuring that the dataset is ready to be trained in our system, we feed our dataset to the bi-RNN with long short-term memory (LSTM) cells (Hochreiter and Schmidhuber, 1997) and an attention mechanism (Luong et al., 2015) along with the word embeddings we extract from the dataset and trained everything jointly. For each different set of hyperparameters, we train the model separately. We save the best three models. Table 3 shows the different hyperparameters used for the three models as well some of the experiments that have been done using GloVe embedding (Pennington et al., 2014) + byte pair encoding (BPE) (Sennrich et al., 2015) with a vocabulary of 10K sub-word units (Experiment-1), GloVe + without BPE (Experiment-2), and the default model. The rest of the hyperparameters are left at their default value.

Models	Batch size	Dropout	BPE	Embedding
Default	64	0.3	Yes	GloVe
Experiment-1	64	0.4	Yes	GloVe
Experiment-2	64	0.3	No	GloVe
Model-A	32	0.6	No	BERT
Model-B	100	0.7	No	BERT
Model-C	182	0.7	No	BERT

Table 3: Different hyper parameters of the three models.

We also experiment with the celebrated Transformer mode (Vaswani et al., 2017). However, this model results in very low BLEU scores when evaluated on the dev set. Moreover, it takes about four days to finish training in one experiment. So, we decide to exclude it from further consideration.

3.4 Model Ensembling

Before the test set is released, we train different models using the training set and evaluate them

[2] https://github.com/google-research/bert

[3] https://opennmt.net/OpenNMT-py/options/train.html

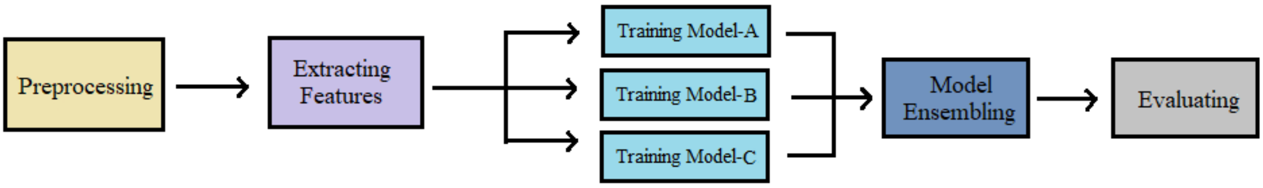

Figure 1: Flowchart of our system.

using the dev set. After training our system, we choose the best three models and ensemble them to get the final output.

4 Results

The results based on the dev set are show in Table 4. The table shows the results of our three models, which we choose for the ensembling step, as well as the other experiments mentioned earlier. The Table shows the difference between them using the BLEU score. From the above table we can notice that training without using BPE improves the results. Moreover, we have chosen the pre-trained BERT because it improves the results compared to the GloVe embedding.

Models	BLEU
Default	32.99
Experiment-1	34.80
Experiment-2	35.21
Model-A	36.88
Model-B	37.07
Model-C	40.93

Table 4: Results of our experiments for the dev dataset.

For evaluation on the test set, we combine the train and dev dataset of each group into one file. Table 5 shows the number of examples in each group after combining them into one file.

	Agent	Customer
Combined train + dev	8,669	7,077

Table 5: Number of examples after combining the files.

We train each group separately and then we ensemble the three models into one. This model is used to get the target of each sentence in the test set of each group. It is worth mentioning that we only use the small dataset provided with the shared task.

Table 6 shows the results for the human evaluation between the human, best score and our model for the English→German scores.

Team	Agent Ave.
Human	91.43
Best	88.21
Our Model	63.93

Table 6: Results of the human evaluation.

Table 7 shows the results we get in the shared task compared to the baseline and the best results. We can see that the agent BLEU score of our model is higher than the baseline, which is translating from English to German. On the other hand, the customer BLEU score for the baseline beat our model, which is translating from German to English.

5 Conclusion

This work describes JUST's submission to the WMT20 chat translation task. For all two translation directions, English→German and German→English, we used the pre-trained BERT embedding with the bi-RNN. We trained one model with different hyperparameters and then ensembled to one final system to translate the test set provided by the shared task. At the end of this work, we find out that a simple NMT model with BERT embedding can achieve surprisingly good results even if it is trained on a very small corpus.

References

Malak Abdullah and Samira Shaikh. 2018. Teamuncc at semeval-2018 task 1: Emotion detection in english and arabic tweets using deep learning. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 350–357.

		Agent BLEU	Customer BLEU	Agent TER	Customer TER
Best	-	60	62	0.25	0.23
FAIR-WMT19	Baseline	43.4	49.7	0.379	0.3195
test1_corpus	Our model	46.4	42.5	0.382	0.4015

Table 7: Results of the shared task.

Ghadeer Al-Bdour, Raffi Al-Qurran, Mahmoud Al-Ayyoub, and Ali Shatnawi. 2019. A detailed comparative study of open source deep learning frameworks. *arXiv preprint arXiv:1903.00102*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, and Kyu-Young Kim. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. *arXiv preprint arXiv:1909.05358*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adolfo Hernández. 2009. A ngram-based statistical machine translation approach for text normalization on chat-speak style communications.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Lstm can solve hard long time lag problems. In *Advances in neural information processing systems*, pages 473–479.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

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.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

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

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200. Cambridge, MA.

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.

Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398.