Training a transformer with BERT embeddings: a project report

Matt Twete (mtwete@pdx.edu), Russell Scheinberg (rschein2@pdx.edu)

Abstract

We set out to challenge the state of the art by building a Japanese-specific neural machine translation (NMT) model, and this paper is a concise of our NLP adventures. We give a thorough description of our search of current research in sentence-level Japanese-English NMT systems. We then describe the final model that we implemented, a transformer model trained with contextualized encodings derived from a Japanese BERT model. In the course of this report we will describe our research process, decisions made, and results obtained. While our final results are not ideal, we hope to continue work in the future to get a more desired outcome.

1 Introduction

While massively multilingual models have shown promising results in recent years, the authors, both speakers of Japanese and English, were intrigued by the question of whether a more targeted approach that takes advantage of the idiosyncrasies of a single language pair like Japanese-English could achieve results superior to those of the large multilingual models.

What then are the characteristics of the language pair, Japanese-English, that we chose? The two languages of the pair are genetically distant, and exhibit a number of differing traits. These include the topic-comment structure of Japanese, as opposed to the more familiar subject predicate of European languages like English; a complex syllabic writing system (comprising hiragana, katakana and kanji or Chinese characters) as opposed to the alphabetic writing systems of European languages (most commonly the latin alphabet); and also a prevalence of homonyms (words that are pronounced the same) and homographs (words that are written the same) in Japanese. As a simple example, the Japanese word (rai-jin-gu) borrowed from English can be a term meaning ”rising” or ”writhing” (as in a ”writhing test” for testing the reactions of small animals to drug testing), so disambiguation of homonyms/homographs can be said to be an important issue in Japanese-English translation.

Thus, the first stage of our project was to examine the previous work done in the area of Japanese-English sentence-level NMT systems to see if we could build on previous work and advance the state of the art. While the specific goal of a bilingual translation model may seem of limited interest, our underlying hypothesis was that taking into account specific linguistic attributes of a language pair could lead to improved performance.

2 Related Work

Previous work in sentence-level Japanese English NMT systems takes several approaches to improve on the state of the art. We will briefly review three approaches which are relevant to the present work: dealing with lexical ambiguity, incorporation of syntactic information, and reordering of source-target sequences to improve translation between these structurally disparate languages.

2.1 Resolving Lexical Ambiguity

D1 *“Resolving Lexical Ambiguity in English-Japanese Neural Machine Translation”* (Do, Zeng, and Paik 2021)

The authors address the problem encountered in the original Transformer model (Vaswani et al. 2017) of lexical ambiguity in Japanese translation by integrating a pretrained BERT language model (Devlin et al. 2019) into a transformer. Specifically, they describe training two architectures, one of which utilizes context-sensitive word embeddings derived from a pretrained Japanese BERT model, and the other of which replaces the entire encoder component of a vanilla Transformer with the BERT model. Do, Zeng, and Paik then demonstrated improved performance on a subset of the IWSLT 2017 Japanese-English dataset that contains homographs, as well as improved the Bilingual Evaluation Understudy or BLEU scores(Papineni et al. 2002) on JESC, the Japanese-English Subtitle Corpus. However, the first of the two models performed best.

D2 *"Point, Disambiguate and Copy: Incorporating Bilingual Dictionaries for Neural Machine Translation"* (Zhang et al. 2021)

The authors propose a novel architecture for incorporating a bilingual dictionary into a Transformer by introducing three components, a Pointer, a Disambiguator and a Copier that are all integrated into the basic Transformer. The Pointer "points to" source words that are candidates for using dictionary definitions. It does this by utilizing the source encoding layer to obtain candidate source word encodings, and novelly combining the semantic information of the translation candidates as well. The Disambiguator module uses both source-view and target-view disambiguation since the correct sense of a source word and the translation of that sense of the source word should fit both contexts. Finally, the Copier module is responsible for combining the results of the other two modules into a probability distribution for adopting a given dictionary translation as the translation of a given source word. The Point, Disambiguate and Copy architecture is attractive in that it incorporates professionally assembled linguistic resources (a bilingual dictionary) into the probability distribution models in the encoder and decoder of a transformer, and Zhang et al. 2021 do achieve SotA BLEU scores on the ASPEC dataset.

2.2 Incorporating Syntactic Information

D3 *"Neural Machine Translation with Synchronous Latent Phrase Structure"* (Harada and Watanabe 2021)

In order to incorporate syntactic information into NMT, it is necessary to have a low-cost syntactic parser, as large-scale human annotation would be prohibitively expense and of questionable value. (Harada and Watanabe 2021) first induce the phrase structure of a source sentence and of candidate target sentences on the basis of the syntactic distance introduced by Shen et al. 2018, then shape the loss function in the cross-attention layer so that the source and target syntactic distances match. Harada and Watanabe 2021 find that this synchronization of latent phrase structure improves BLEU scores over the SotA.

D4 *"Improving Neural Machine Translation by Transferring Knowledge from Syntactic Constituent Alignment Learning"* (Su et al. 2022)

Su et al. 2022 incorporate a GAN, with the discriminator trained to discriminate "true" (ground truth) translations from "false" (randomly selected) translations by extracting and aligning constituents of source and target translations.

2.3 Reordering of Source Language Segments

D5 *"Preordering Encoding on Transformer for Translation"*(Kawara, Chu, and Arase 2021)

This paper proposes two methods for reordering the words of the source text fed to the transformer so as to better match the probable reordering of the words in the target language.

D6 *"Explicit Reordering for Neural Machine Translation"* (Chen et al. 2020)

This paper improves transformer translation results on the ASPEC dataset by a "reordering fusion-based source representation" (reFSR), which includes both the original word order information present in the source text, as well as reordered word order information generated by an encoder. The reordering is a 'global' reordering that takes into account the sentence length.

	Datasets				
	EN-JP				JP-EN
Ref.	JESC	IWSLT	KFTT	ASPEC	ASPEC
D1	20.31	8.67	-	-	-
D2	-	-	32.18	-	-
D3	-	-	-	-	29.79
D4	-	-	-	42.13	-
D5	-	-	-	-	25.28
D6	-	-	-	-	33.27

Table 1: BLEU Scores of the State of the Art from the Related Work

3 Methodology

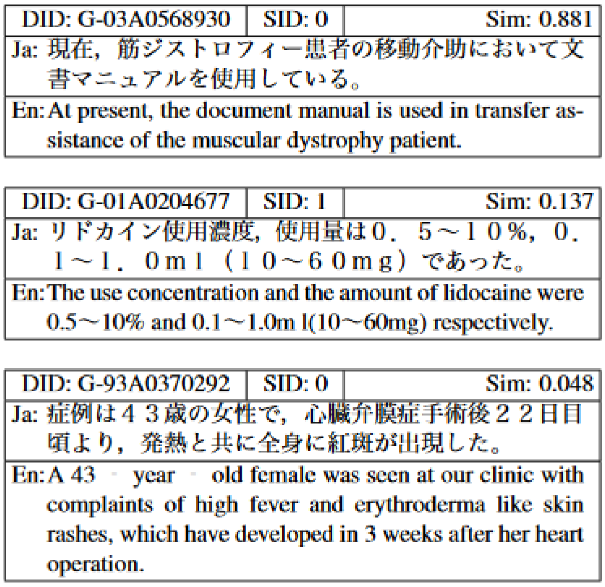
3.1 Dataset

As can be seen in table 1, the Asian Scientific Paper Excerpt Corpus (ASPEC) dataset (Nakazawa et al. 2016) is the most commonly used parallel Japanese-English dataset suitable for NMT training and evaluation. The dataset comprises a corpus of 3 million parallel sentences (ASPEC-JE) from technical and scientific papers originally authored in Japanese and translated manually into English.

The dataset quality is somewhat suspect, which was confirmed after inspection by one of the authors of the present paper. This is in part because the English ”translations” are actually found and matched automatically based on a similarity score from Utiyama and Isahara 2007. ASPEC is sorted based on this similarity score, so the higher quality translations are found in the first million entries, and none of the papers from the previous work section that use ASPEC utilize the entire dataset. The paper introducing the dataset curiously does not involve asking translation experts to rate the quality of the parallel translations provided. Although flawed, the Japanese government’s sponsorship of this large-scale parallel bilingual corpus remains a common reason that it is used as a benchmark for Japanese-English NMT tasks.

The following is a sample of entries from ASPEC (Nakazawa et al. 2016), with the similarity score indicated.

Figure 1: Examples sentences from ASPEC dataset



DID: G-03A0568930	SID: 0	Sim: 0.881
Ja: 現在，筋ジストロフィー患者の移動介助において文書マニュアルを使用している。		
En: At present, the document manual is used in transfer assistance of the muscular dystrophy patient.		

DID: G-01A0204677	SID: 1	Sim: 0.137
Ja: リドカイン使用濃度，使用量は０．５～１０％，０．１～１．０ｍｌ（１０～６０ｍｇ）であった。		
En: The use concentration and the amount of lidocaine were 0.5～10% and 0.1～1.0m l(10～60mg) respectively.		

DID: G-93A0370292	SID: 0	Sim: 0.048
Ja: 症例は４３歳の女性で，心臓弁膜症手術後２２日目頃より，発熱と共に全身に紅斑が出現した。		
En: A 43 year old female was seen at our clinic with complaints of high fever and erythroderma like skin rashes, which have developed in 3 weeks after her heart operation.		

3.2 Model

For our project we implemented a variation of the BERT-WE model used in Do, Zeng, and Paik 2021. We were originally hoping to combine this work with the work done by Harada and Watanabe 2021, but were unable to due to the code not being available and the difficulty associated with implementing it ourselves. The model that was implemented was essentially a vanilla transformer, the key difference was that a pre-trained Japanese BERT model was used to generate contextualized word embeddings fed into it. Some changes were made from the standard transformer configuration, the embedding layers and all sublayers’ output dimension, and the dimension of the inner layer in every fully connected feed-forward layer were set to 768 and 3072 respectively to match the hidden size of the pre-trained BERT model. Additionally, the two differences from Do, Zeng, and Paik 2021 are that instead of 3 encoder and decoder blocks, we used 6 encoder and decoder blocks in the transformer to see if the increased model capacity results in significantly different performance and the language direction was Japanese to English instead of English to Japanese. The transformer architecture, training logic and evaluation logic was implemented in PyTorch using an online resource that is cited in the source code.

The Japanese BERT base model used was downloaded from Hugging Face using their convenient AutoModel library, in this case AutoModelForMaskedLM was used. The Japanese BERT model shared the same architecture used in the original BERT base model having 12 layers and attention heads with a dimension of 768 for the hidden states. It was trained on the Japanese version of Wikipedia containing approximately 30M sentences. The weights of the pretrained BERT model were not updated during training. To generate the word embeddings used by the model, the representations of the source tokens in the hidden state of the last 10 layers of the BERT model were extracted and averaged. This logic and all other parts of the code were written by the authors.

3.3 Tokenizers

The Japanese tokenizer used in this work is the pretrained Japanese BERT tokenizer which was also downloaded from Hugging Face using their AutoTokenizer library. This was needed so that source sentences could be tokenized in the correct way in order to be fed into the Japanese BERT model to get the contextualized word embeddings. No changes were made to this tokenizer.

The English tokenizer was a English BERT tokenizer that was trained on the training data used for the model. This was also done with the AutoTokenizer library from Hugging Face. An English BERT tokenizer was chosen mainly to allow for the possibility of running experiments with English as the source language in the future, but also out of convenience as the Hugging Face libraries are easy to use. Once training was complete the tokenizer was saved to a file for easy reuse.

The training, test and validation data was tokenized using the two tokenizers and then saved to files as torch tensors so that they could be loaded directly instead of having to be tokenized every time.

3.4 Coding Environment

All of our implementation was done using Google Colab[1]. This allowed for easy access to Google Drive as well as GPUs for training. A Google Colab Pro+ subscription was purchased so that high performance and high RAM GPUs could be used. Additionally, in order to store model checkpoints a higher capacity Google Drive subscription was purchased otherwise storage space would have run out. We plan to upload the notebook to GitHub in the near future.

4 Experiments

Due to time and compute restraints, experiments were limited to training a single model with the architecture described in the previous section.

4.1 Training Data

The first 1M sentences in the ASPEC dataset were used for training, this resulted in only the highest quality sentence pairs being utilized. However, the full test and validation sets, with 1790 and 1812 parallel sentences respectively, were used. The BLEU score was used for evaluation on the test set.

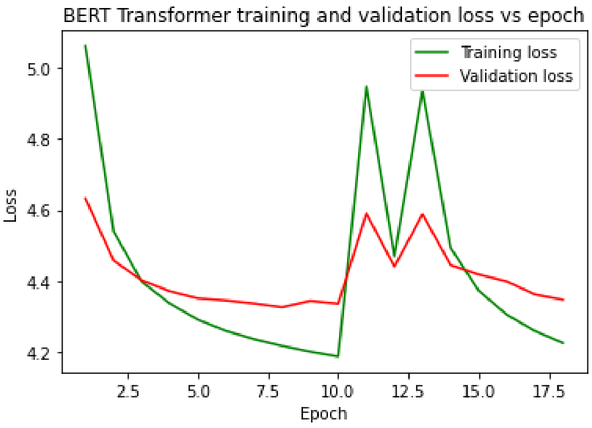
4.2 Training Details

All training was done on Google Colab using primarily a NVIDIA A100 GPU. The Adam optimizer was used with an initial learning rate of 0.0001 and betas set to 0.9 and 0.98 respectively. The model was evaluated on the validation set after every epoch to monitor training. The model was trained for 18 epochs taking approximately 42 hours total, training was stopped at this point due to compute limitations. Training had to be restarted at the 11th and 13th epoch due to notebook run time limitations and an error with Google Colab respectively.

5 Results

The plot of the training and validation loss versus training epochs can be seen below.

Figure 2: Training and validation loss over training



You can see that initially training proceeded as expected, there was a sharp decrease in the training loss and to a lesser degree the validation loss over the first 10 epochs. Then there is a sudden spike at the 11th epoch, which is likely due to some issue related to restarting training from a checkpoint after the 10th epoch. The loss decreases again until another training restart at epoch 13 where it jumps up another time. Finally the loss decreases after the second jump until the end of training. This clearly indicates there was an issue loading the models from checkpoints. When it came to evaluation on the test set, unfortunately the model had a BLEU score of 0.0. This was the same for the model at both epoch 8 and 18, with epoch 8 being checked because it was the last checkpoint before the first training restart. BLEU scores from baseline vanilla transformers trained on varying amounts of the ASPEC data from the related works section can be seen in table 2, obviously beating the results of our model.

Vanilla Transformer BLEU Scores	
Ref.	Test Set BLEU Score
D3	29.48
D5	31.21
D6	23.94

Table 2: Baseline BLEU Scores

To test the translations generated by the model qualitatively, several simple sentences were fed to

[1] The link to our openly available Google Colab notebook can be found here: https://colab.research.google.com/drive/1Cu-4-Uc5ZHL-S_nwaW8HUt7jH1v2wiJH?usp=sharing

the model to check the generated translation. The results can be seen in the table 3. It is clear the model is not accurately translating the sentences, in fact two totally different input sentences resulted in the same translation. It appears the model is memorizing the training data to some degree as the translations are very similar to sentences in the training set. This is odd given that there were not signs of severe overfitting in the figure 2. We are not sure exactly why the model generates such kinds of translations.

6 Interesting Insights

Clearly the model did not train as expected and the results were not what we hoped. However, we did learn useful insights regarding neural machine translation and research in general. For one, some of the common libraries, OpenNMT and FairSeq, are not particularly easy to use and even installing or running very simple scripts can lead to issues. Additionally, it would be difficult to implement highly customized models like the one described in this paper in those libraries. However, Hugging Face libraries were very easy to use and had great documentation. Also, we learned that it can be difficult to get code from papers you are trying to build on. The training of the model did not work as we expected when dealing with restarting from checkpoints, we saw a sharp increase in the loss every time training restarted for unclear reasons. If we had more time we would investigate the causes of the poor translation, it could possibly be the result of the greedy-decoding function used or maybe even how the embeddings were extracted from the BERT model. And if the models start working properly, then it would be interesting to experiment with training two models, one for each language direction, to see how the results compare. Or possibly even using a multilingual BERT model instead of language specific ones.

7 Conclusions

In this work we attempted to implement a variation of the model described in Do, Zeng, and Paik 2021, but with a different language direction, model size and dataset. The hope was that the contextualized word embeddings would result in higher quality translations overall and especially for sentences with lexical ambiguity. Although we did not get any usable results, we still gained significant insight into neural machine translation and the process of conducting research. Additionally, even though the results were not what we hoped, we were able to get a large model training on a moderately large dataset in Google Colab which is not easy in of itself. We learned about the key libraries used for these kinds of tasks and some of the recent research that is going on in this field. We hope to resolve the issues that led to the poor results in the future and eventually get a working model trained.

8 Ethical Considerations

We detected minimal ethical considerations in our project.

Regarding the dataset, ASPEC is made up of articles intended for publication, and we received permission from the Japanese government to use the dataset. The publications used to create the ASPEC dataset were authored in the Japanese language, primarily by Japanese nationals, and presumably with Japanese government support. So the scientific research they constitute may represent a Japanese national orientation. The ASPEC dataset was checked partially by crowdsourced workers, a practice which can carry with it some risk of exploitation or bias, though bias is unlikely to be relevant in checking the translation of technical texts.

If the model had worked as intended, there would have been some possibility of misuse if it was made publicly available. For example, someone could use it to translate hate speech into English from Japanese. However since the model did not train properly and the weights of the trained version are not publicly available, we find that there is little potential for misuse. Additionally the people who would most likely benefit from the model would be Japanese and English speakers, speakers of other languages as well as people without access to computers or the internet would likely not benefit much from it.

Regarding our use of Colab, the computation resources that we used were not more than would be expected from students in an NLP class, although Google does not seem to publish a statement regarding ethical considerations for its practices running Colab, except to forbid certain activities like training Deepfakes.

References

Papineni, Kishore et al. (2002). “BLEU: a method for automatic evaluation of machine translation”.

Example output from trained model	
Source Text:	私の名前は田中です。
Meaning:	My name is Tanaka.
Model's translation:	The case was a 66 - year - old woman.
Source Text:	バナナが好きです。
Meaning:	I like bananas.
Model's translation:	The case was a 66 - year - old woman.
Source Text:	大学で物理学を勉強している。
Meaning:	I am studying physics at university.
Model's translation:	The patient was discharged from the hospital on day 14.

Table 3: Example translations generated by model at epoch 8

In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pp. 311–318.

Utiyama, Masao and Hitoshi Isahara (2007). "A Japanese-English patent parallel corpus". In: *MT-SUMMIT*.

Nakazawa, Toshiaki et al. (May 2016). "ASPEC: Asian Scientific Paper Excerpt Corpus". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 2204–2208. URL: https://aclanthology.org/L16-1350.

Vaswani, Ashish et al. (2017). "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., pp. 6000–6010. ISBN: 9781510860964.

Shen, Yikang et al. (July 2018). "Straight to the Tree: Constituency Parsing with Neural Syntactic Distance". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 1171–1180. DOI: 10.18653/v1/P18-1108. URL: https://aclanthology.org/P18-1108.

Devlin, Jacob et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: https://doi.org/10.18653/v1/n19-1423.

Chen, Kehai et al. (2020). "Explicit Reordering for Neural Machine Translation". In: *ArXiv* abs/2004.03818.

Do, Quang-Minh, Kungan Zeng, and Incheon Paik (2021). "Resolving Lexical Ambiguity in English-Japanese Neural Machine Translation". In: AICCC 2020. Kyoto, Japan: Association for Computing Machinery, pp. 46–51. ISBN: 9781450388832. DOI: 10.1145/3442536.3442544. URL: https://doi.org/10.1145/3442536.3442544.

Harada, Shintaro and Taro Watanabe (Aug. 2021). "Neural Machine Translation with Synchronous Latent Phrase Structure". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*. Online: Association for Computational Linguistics, pp. 321–330. DOI: 10.18653/v1/2021.acl-srw.33. URL: https://aclanthology.org/2021.acl-srw.33.

Kawara, Yuki, Chenhui Chu, and Yuki Arase (2021). "Preordering Encoding on Transformer for Translation". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29, pp. 644–655. DOI: 10.1109/TASLP.2020.3042001.

Zhang, Tong et al. (Aug. 2021). "Point, Disambiguate and Copy: Incorporating Bilingual Dictionaries for Neural Machine Translation". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and* *the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, pp. 3970–3979. DOI: 10.18653/v1/2021.acl-long.307. URL: https://aclanthology.org/2021.acl-long.307.

Su, Chao et al. (Apr. 2022). “Improving Neural Machine Translation by Transferring Knowledge from Syntactic Constituent Alignment Learning”. In: *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 21.5. ISSN: 2375-4699. DOI: 10.1145/3510580. URL: https://doi.org/10.1145/3510580.