CS605 – Natural Language for Smart Assistants

Group Project Report

Shopee Product Title Translation

Group 12
Lin Tzu-Chun
Low Yee Lee
Tang Yue
Zheng Wanyu

Abstract

Machine translation has been widely employed for real world translation tasks nowadays, with the benefits of faster translation speed, consistent terminology, high productivity, and lower translation cost. This project aims to study and compare machine translation models ranging from statistics machine translation (SMT) to neural machine translation (NMT) and extends the study to incorporate with pretrained model.

Introduction

Shopee as one of the most rapidly developing e-commerce platforms in Southeast Asia (SEA), is actively looking for opportunities to expand its business to the global market. With the increasing varieties of products and services from non-English speaking countries entering to foreign markets, the demand of translating product information to English is rapidly growing. Considering the limited capacity and productivity of human translators, leveraging on machine translation technology to fulfil the expanding business demand becomes a more applicable solution. In year 2020, Shopee posted Shopee Code League with one of the subtasks being Product Title Translation (from tradition Chinese to English) and the datasets used were later made public on Kaggle.com. This project is motivated by the emerging demand and growing market of machine translation in business, and targets to study and compare traditional machine translation models and neural machine translation models, namely in terms of effectiveness and efficiency.

Approach

Following the generic approach of natural language processing (NLP), we first pre-processed the text to keep only necessary information. An additional step before model building is data parallelization, by which the unpaired data are aligned to be parallel pairs. Afterwards, a SMT model was built as an evaluation baseline where translations are generated on the basis of statistical models, followed by NMT models such as Seq2Seq with attention and Long Short-Term Memory (LSTM). Lastly, we attempted to extend the study to incorporate with pretrained model into NMT.

Data Pre-processing

The data provided in our project is bilingual, Tradition Chinese and English. This dataset is made up of three subsets: training dataset, validation dataset and test dataset. Both the training data and validation data are bilingual and stored separately. Test data contains is in Traditional Chinese for further product title translation and test purpose, it only has one column “text”. And there are two columns in training data: product_title and category. The dataset details could be seen as below:

Table 1: Dataset Details

Data Set	Language	Data Size	Number of Column
Training Data	Traditional Chinese	500,000	2
	English	499,999	
Validation Data (dev)	Traditional Chinese	1,000	1
	English		
Test Data	Traditional Chinese	10,000	1

In raw data, there are a lot of noises in product title. For better translation results, data pre-processing would be the first step in our project. Since the product title is bilingual, we use Jieba package for Traditional Chinese pre-processing and NLTK package for English pre-processing separately. The text pre-processing steps for bilingual data are shown in the below table.

Table 2: Data Pre-processing steps

No	Traditional Chinese Data	English Data
1	Null value removal	Null value removal
2	Punctuation removal	Punctuation removal
3	Non-Chinese words removal	Non-English words removal
4	Lower casing	Tokenization
5	Tokenization	Lemmatization

Data Parallelization

The problem we faced with the training data is that there it is not paired up with the Traditional Chinese words in parallel order with the target English words. The approach taken here is to make use of the Google Translate API, converting the original Chinese words into English, followed by using cosine similarity to match the English output to the given English set in the training data. However, the given English subset has too much noisy characters and the matched scores of cosine similarity are poor resulting in low quality of parallel Traditional Chinese-English corpus. The workaround solution employed is to make use of the Google API translated English output as the training target label instead of using the given English training subset. While waiting for the generation of the translated parallel corpus, a few experiments were carried out using the given paired dev set initially. In the end, about 130k lines of parallel Traditional Chinse-English product titles are generated for use in this limited project constrained by time and computing power.

Statistical machine translation (SMT)

Statistical machine translation (SMT) combines a translation model with a target language model to convert sentences from the source text in one language to sentences in the target language. This is illustrated in the following diagram. The translation model maps words and phrases from the source language to the target language. The language model captures statistics of how likely words follow a specific sequence in the target language. SMT, therefore, tries to maximize the probability of choosing a target sentence that is the translation of the source sentence. SMT is outperforming conventional rule-based translation between source-to-target language pairs where the word order is similar, it enables us to construct robust translation systems with low cost in short development cycles if the training data are available.

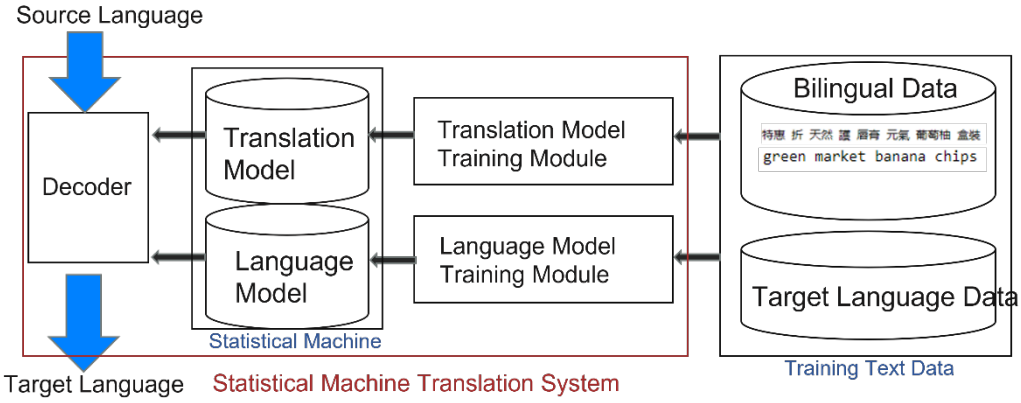

Figure 1: Statistical Machine Translation Architecture

Neural Machine Translation with Attention Mechanism

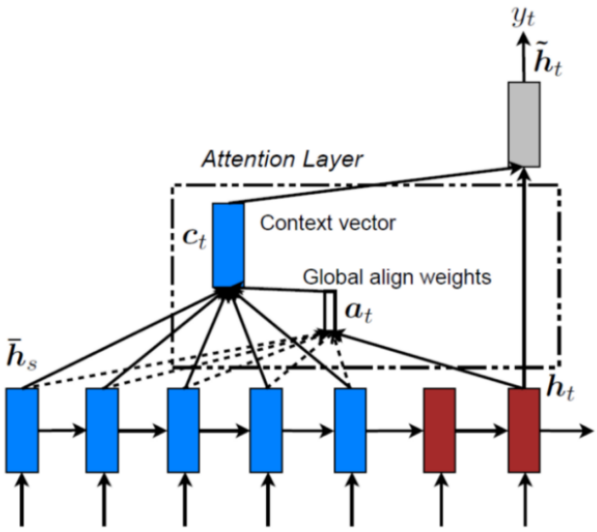

Figure 2：Seq2seq with Attention Mechanism

In NMT, we are making use of deep learning and representation learning that is based on a recurrent neural network (RNN), encoder-decoder structure to do word sequence modelling, also known as Sequence-to-Sequence (Seq2Seq). The use of vector representations for words and internal hidden states, allows the encoder to encode a source sentence into hidden states and updates it based on its inputs and previous inputs, for a second RNN, the decoder that takes in the hidden states to predict the output sequence item by item in the target language. The hidden state vector is actually a fixed-length context vector that poses as a bottleneck for these RNN architectures. This can be compensated by an attention mechanism which allows the decoder to focus on different parts of the input while generating each word of the output, allowing for better model prediction.

Neural Machine Translation with LSTM

As the NMT structure, we remove attention mechanism and use Long Short-Term Memory (LSTM) for both encoder and decoder. Encoder processes inputs information and generate context vectors, then decoder uses them and generate outputs. The reason we build this model is to mainly compare with NMT with attention mechanism.

Incorporating Pretrained Models

Adapting pretrained models into machine translation tasks, we attempted to apply the concept of transfer learning that the general language knowledge obtained by pretraining stage could be transferred and applied to finetune stage and downstream NLP tasks. Being trained on large size corpus with deep layers and high dimensional parameters, pretrained models intuitively have a better understanding on language, and it is expected to bring better performance on this machine translation task as it is not limited by the given vocabulary.

Evaluation Method

Bilingual Evaluation Understudy Score, or BLEU for short, is the evaluation matric in our project to evaluate the effectiveness of Machine Translation (MT). A perfect match results in a score of 1.0. BLEU is computed using a couple of n-gram modified precision, specifically,

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Where p_n is the clipped precision for n-gram, N is the number of N-gram, usually N= 4, w_n is between 0 and 1, $\sum_{n=1}^{N} w_n = 1$, usually, $w_n = \frac{1}{N}$. and BP is the brevity penalty to penalize short machine translations.

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases}$$

Where c is the number of unigrams (length) in all the candidate sentences, r is the best match lengths for each candidate sentence in the corpus.
BLEU score usually takes 4-gram, but we chose bigram in this project, which is determined by the text to be translated in dataset. Different from article translation, product title translation was conducted on less coherent sentence. Therefore, it has more requirement on Adequacy and Fidelity but less on Fluency. In this case, bigram is used for BLEU score calculation in this project.

Experiment

Statistical machine translation (SMT)

In this project, we use SMT as baseline model to compare with NMT model and use the BLEU score as evaluation matrix. Here we chose IBM lexical alignment models, which are simple statistical translation models. These models take a collection of alignment pairs between the source and target languages and compute probabilities of their associations or alignments. The nltk.translate package provided the implementation of the IBM alignment models. The basic IBM Model 1 was used in project, which performs a one-to-one alignment of the source and target sentences. Thus, the simple SMT model could produce exactly one target word for each source word without considering any reordering or translation of one source word to multiple words or the dropping of words.

Firstly, we import paired dev_set which contains Traditional Chinese and corresponding English translated product title. The Aligned Sent class in nltk.translate package would be used to alignment the Traditional Chinese and English data. Next, we leverage IBMModel1 to train paired training data. Then we create the list of tradition Chinese – English sentence with pairs with AlighedSent for model training. The translation_table of IBM1 model stores the probabilities of an alignment between a given Traditional Chinese and the corresponding English words, then the model would pick the English word that is more likely a translation of the given Traditional Chinese words using the alignment probabilities. Running the learnt translation table to do prediction on the test set gives us 9361 rows of output out of 10,000 rows. Then the same translation_table is used to run the Bleu evaluation on paired dev_set so as to make use of the given English labels as references to achieve Bleu score of 3.38 (n-gram=2, non BP).

Next, the custom generated paired training data by Google Translate API is loaded in, the model translation_table learning task takes up a training time of around 20 minutes in total. It is remarkable that SMT training time is faster than NMT model, with prediction on test set giving a higher output proportion 9825 out of 10,000 rows now. The Bleu evaluation using same dev_set gives an improved score of 5.18 (n-gram=2, non BP).

Neural Machine Translation with Attention Mechanism

The next model we tried out is the Seq2Seq with attention mechanism model. Pre-processing of the Traditional Chinese and the English words in paired training set are needed to convert them into word token embeddings, as well as adding ‘<start>’ and ‘<end>’ tokens to the word sequences so that the model can recognise the end of sequence translation. Initial experiment on using the paired dev_set of

1000 rows of examples, the vocabulary input size of 3224, training parameters used were 256 dimensions for input embedding later and 1024 hidden state units. The encoder is defined with the embedding layer and followed by the GRU layer with sigmoid as activation function, same as the decoder with addition of the attention mechanism using Bahdanau Attention, a form of additive attention to give attention score:

$$\text{score}(s_t, h_i) = v_a^T \tanh(W_a[s_t; h_i])$$

where ***v*** is the learnt weight through the fully-connected layers, ***s*** is the decoder hidden state (query vector) concatenate with ***h,*** the encoder hidden state (value vector). The attention weights are then calculated as a softmax across the attention scores and then context vector attained as a weighted sum of the encoder outputs hidden states. After defining the Adam optimizer and Cross entropy loss function, training run of 100 epoch done, with final loss obtained at 0.0312 and total time duration of less than 5 minutes.

Next experiment carried out is loading in the paired parallel data from Google Translate API as training data. The input vocabulary size is around 150k from the 130k lines of parallel training data. Because of this large vocabulary input size, to circumvent the out-of-memory errors, the input embedding layer has to be restricted to a small dimension size of 32, number of hidden state units to 32 and training batch size set at 128. The resulting loss after 100 epochs attained is 0.0914, total time duration taken is 14 hours. It is obvious that to achieve this better translation performance using NMT has this high trade-off of long training time compared to the SMT model.

Neural Machine Translation with LSTM

To emphasize the importance of attention mechanism, we attempted to build another NMT model with LSTM. Firstly, we imported pre-processed and tokenized data, then we one hot encoding for word embedding. Because of limitations of Colab, the training process just utilized 2000 rows of data. English vocabulary size is around 9000 and Chinese is around 4000, training parameters used were 256 dimensions for input embedding later and 50 epochs. The model consists of an embedding layer, LSTM for encoder, LSTM for decoder, and a fully connected layer with SoftMax.
Although the model was done, BLEU score is quite low. The reason probably is from the embedding layer. One hot encoding may not be optimal for a large size dataset and can easily cause memory ran out issue. Due to computational resource limitation, we discontinued the training of this model.

Seq2Seq Model with Pretrained Model

To incorporate pretrained models into NMT model, two methods were discussed: the first is to use pretrained model with a language model head on top, and the second is to use pretrained model to initialize both encoder and decoder of the previously build Seq2Seq with attention model. The second approach was selected as it is able to provide an apple-to-apple comparison of the translation results with or without pretrained models. Recall that in the previously built NMT model, Seq2Seq with attention mechanism, the text was converted to word token embeddings and ‘<start>’ and ‘<end>’ tokens were added to word sequences. Here, we simply take the same Seq2Seq model architecture, but encode and decode the text with pretrained encoders, namely ‘xlm-roberta-base’ in this case. ‘xlm-roberta-base’ was selected because it was trained on a corpus of 100 languages, and it is able to take care of both traditional Chinese and English. ‘xlm-roberta-base’ encodes text sequence differently from the word token embedding employed before. Instead of adding the ‘<start>’ and ‘<end>’ tokens to indicate beginning and end of a sequence, encoder_plus function of ‘xlm-roberta-base’ encoder automatically detects the start and end of a sequence and adds '<s>' and '</s>' tokens respectively. The

rest spaces other than text are encoded by a special '<pad>' token. Having built up the flow and pipeline of this end-to-end machine translation model, the training and prediction failed to be carried out due to limited computational resource. The vocabulary of 'xlm-roberta-base' has a size of more than 250000. When an experimental trial was carried out with only 21 row inputs, encoding step takes about 3min wall time and training for 1 epoch takes about 45min. As such, we discontinued the training of this model.

Results

In this project, we leveraged four models to translate product title from Traditional Chinese word into English, they are Statistical Machine Model (SMT), Seq2Seq with Attention Model, LSTM and Seq2Seq with Pretrained model respectively. As the computing limitation, we take Seq2Seq with Attention Model as our translation model. Here we mainly compared with baseline model and Seq2Seq with Attention Model. Bleu score was taken as evaluation matrix to compare models' performance, which could be seen as below.

Table 3: Model Comparison in terms of BLEU scores and training time

Model	SMT (baseline)	Sequence-to-sequence with Attention
Bleu Score (n=2, without BP)	5.18	19.95
Training time	20 minutes	14 hours

Analysis

Statistical machine translation (SMT)

As baseline model, SMT performs well in translation task with Bleu score 5.18. And the total training time is around 20 minutes. The nltk.translate package was used to build SMT model. We can see the availability of SMT, its platforms and algorithms have been done for users and training can be done at a much cheaper rate. As a result, SMT model can train languages very fast in comparison to NMT models. Besides, SMT also requires less virtual space than other NMT models, which make it easier to operate and train. However, SMT relies on probabilistic mathematical theory and the Markov' assumption is weak thus giving a slightly low performance result in comparison with NMT. Another issue is that SMT systems need bilingual corpus that is paired that can be tricky when it comes to finding content written in rarer language. Besides, SMT only can perform well when translating text similar to the training corpuses or domain, otherwise its performance is hardly satisfactory.

Neural Machine Translation with Attention Mechanism

High performing NMT Seq2Seq with attention model giving Bleu Score of 19.95, almost 4 times higher than the score for SMT model shows that the encoder-decoder RNN architecture is suited for language translation tasks. This is aided by the deep learning and representation learning capabilities of the neural networks that can pick up linguistic cues such as the semantics and syntactics that are important for language understanding. Also, present to achieve this aim is the Attention mechanism that provides the decoder with a look at the entire input sequence regardless of the length at every time step, giving the model better chance at understanding long range dependencies of the word tokens. The resulting target output phrases have more fluency, better context awareness, and use of phrase similarities as compared to the probabilistic based, phrase by phrase SMT model. However, all these does not come free but with high computational costs as evident in the long training time.

BLEU Score Evaluation

For all the above results involving BLEU score, the package SacreBLEU is a python library that is widely used by NLP researchers in the community, as quoted "SacreBLEU is currently the de-facto standard for computing and reporting BLEU scores." (Müller, 2020). It is because different tokenization and pre-processing methods give rise to different results that impairs the comparison of model performance. The inputs for the BLEU evaluation in this project are full pre-tokenized sequences unprocessed, whereby using SacreBLEU default settings and left to tokenise on its own. However, we chose to use the n-gram n=2 and brevity penalty = 1 (meaning no penalty imposed), giving the precision score of 2-gram, as our used case problem of product title translation is not predicting long sentences with grammar fluency, and unigram will be deemed too short and isolated, while the standard 4-gram will be too long for short sentences in the case of our examples, hence 2-gram is selected.

Conclusion and Future Work

Seq2seq method is the current state-of-art method for Machine Translation with benefits far surpassing that of SMT, being built on the Recurrent Neural Network, either via LSTM or GRU, more effectively able to extract feature representations such semantics and syntactics that are important for language understanding. Hence, NMT rapidly replaced the SMT model that is obfuscated with probabilities calculation and Markov's assumption. Moreover, with the correctly labelled datasets we can build effective translation machines for different contexts and languages using the same method, requiring much less human engineering effort.

Some limitations of the NMT model exists such as it does not support out-of-vocabulary words as evident in our experiments mentioned above when training is done with the 1k dev_set versus 130k Google Translate custom generated training set. A workaround for this would be to use a large well recognised parallel corpus as training data. However, with large training corpus, the computational costs and training time will be high in order to achieve reasonable model performance.

For future works, we believe that the performance can be improved at lower computational expense using transformers autoencoder architecture. We would want to build the Seq2Seq model with pre-trained bi-directional encoder model such as BERT and a forward-directional decoder such as TransformerXL, to further extend the capability of conditional language modelling of seq2seq and extract the synergy of BERT token masking and bi-directionality for exceptional contextual awareness, together with TransformerXL memory and relative positional encoding for extra fast and accurate prediction.

References

Collins, M. (2011). Statistical Machine Translation : IBM Models 1 and 2.

Cho, K., Gulcehre, C., Bahdanau D., Bougares F., Schwenk H., Bengio Y. (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv:1406.1078

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

TensorFlow, *Neural machine translation with attention*. https://www.tensorflow.org/text/tutorials/nmt_with_attention. Retrieved from URL.

Kezie Shofer, *The Pros and Cons Of Statistical Machine Translation*. Unite Language Group. https://www.unitedlanguagegroup.com/blog/pros-and-cons-statistical-machine-translation. Retrieved from URL.

Luong, M.-T., Pham, H., & Manning, C.D. (2015). Effective Approaches to Attention-based Neural Machine Translation. *arXiv preprint arXiv:1508.04025*.

Marie, B., Kaing, H., Mon, A.M., Ding, C., Fujita, A., Utiyama, M., & Sumita, E. (2019). *Supervised and Unsupervised Machine Translation for Myanmar-English and Khmer-English*. *WAT@EMNLP-IJCNLP*.

Müller, Mathias. (2020, Dec). *Computing and reporting BLEU scores.* https://bricksdont.github.io/posts/2020/12/computing-and-reporting-bleu-scores/. R etrieved from URL.

NLTK 3.6.2 documentation. https://www.nltk.org/api/nltk.translate.html. Retrieved from URL.

Pearse, B. (2019, April). Human and Machine Translation: Both Alive and Kicking — and Here to Stay. *smartcat.com/blog/human-and-machine-translation-both-alive-and-kicking-and-here-to-stay/*. Retrieved from URL.

Zhu, J., Xia, Y., Wu, L., He, D., Qin, T., Zhou, W., ... & Liu, T. Y. (2020). Incorporating Bert into Neural Machine Translation. *arXiv preprint arXiv:2002.06823*.