RNN/CNN-based Neural Machine Translation for Vietnamese and Chinese to English

Rong Feng[1] rf1316@nyu.edu
Xialiang Liu[2] xialiang.liu@nyu.edu
Melanie Tosik[2] tosik@nyu.edu
Ethan Wu[1] yw3375@nyu.edu

[1]Center for Data Science
New York University
New York, NY 10011

[2]Department of Computer Science
Courant Institute, New York University
New York, NY 10012

Abstract

In this paper, we describe the architecture and performance of three neural machine translation systems: a RNN based encoder-decoder without attention, a RNN based encoder-decoder with attention, and a CNN-RNN based encoder-decoder with attention. Each model is trained and tested on two language pairs: Vietnamese to English, and Chinese to English. We find that for Vietnamese-English, the RNN encoder-decoder with attention performs best, with a BLEU score of 25.37 on the Vietnamese test set. For Chinese-English, the best model architecture is the CNN-RNN based encoder-decoder with attention. The corresponding BLEU score is 22.10 on the Chinese test set. The code is available from the project GitHub repository[1].

1 Introduction

Sequence-to-sequence models are powerful tools for a multitude of tasks in natural language processing. General purpose encoder-decoder frameworks have successfully been applied to text summarization (Nallapati et al., 2016), conversational modeling (Vinyals and Le, 2015), video captioning (Venugopalan et al., 2015), and even interpreting dialects of Python (Zaremba and Sutskever, 2014).

Sequence-to-sequence models were particularly transformative in machine translation (MT). Here, the ability to map arbitrary-length input sequences to arbitrary-length output sequences using a fixed-size architecture is crucial because a simple mapping from a single input token to the correct output token rarely exists due to the ambiguity of language.

Statistical machine translation (SMT) systems aim to overcome this limitation by explicitly maximizing the probability $\mathbb{P}(S|T)$, i.e. choosing the sentence S that is most probable given T, thereby minimizing the chance of error (Brown et al., 1990). This specification formalizes the notion of a number of candidate translations, along with the need for a decoder to select the most likely translation from the model's output probability distribution (Williams et al., 2016). Typically, practical implementations of SMT are phrase-based systems that translate sequences of words as atomic units (Koehn et al., 2003). While conventional SMT can be effective, it suffers from one major shortcoming: translation is not a token-level function and requires (at least) the context of the entire sentence.

Neural machine translation (NMT) is an end-to-end learning approach that offers a key advantage over phrase-based translation systems. Instead of tuning several specialized sub-components, NMT only requires a single large neural network model that can both read an input sentence and predict a correct translation. Specifically, NMT relies on recurrent neural network (RNN) models that are arranged in an encoder-decoder fashion. As described in Bahdanau et al. (2014), the encoder first reads and encodes a source sentence into an internal fixed-length representation called the context vector. The decoder then outputs the translation based on the context vector. The entire system is jointly trained to maximize the conditional probability of a correct translation given the input sentence.

[1]https://github.com/ds1011teamproject/translation

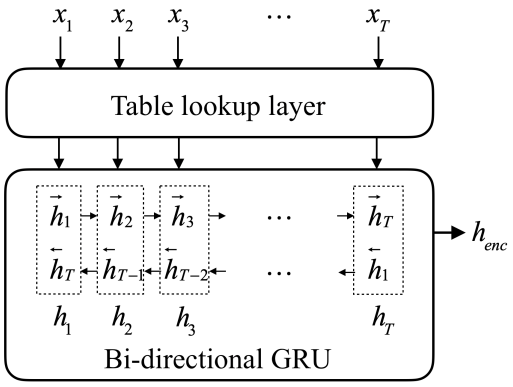

Figure 1: Encoder structure of `RNNbase`.

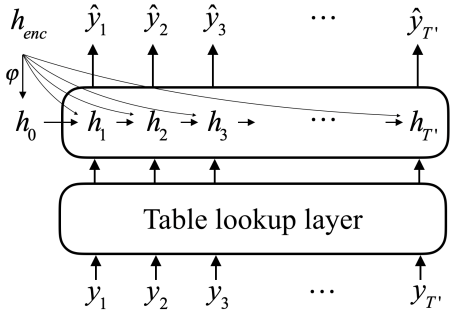

Figure 2: Decoder structure of `RNNbase`.

A common issue when training RNN models is learning long-term dependencies. The solution is the addition of the attention mechanism (Bahdanau et al., 2014), which allows the model to search for parts of the source sentence that are relevant to predicting the target word at the current stage of decoding.

In this paper, we compare three variants on Vietnamese-English and Chinese-English translation systems. First, we implement a RNN-based encoder-decoder model, denoted by `RNNbase`. Next, we introduce a RNN-based encoder-decoder model with attention, denoted by `RNNattn`. Finally, we replace the RNN encoder with a convolutional neural network (CNN) in our third model, `CNNattn`.

2 Model architecture

2.1 RNN encoder-decoder without attention

The first model we implement is a RNN-based encoder-decoder system with gated recurrent units (GRU), with a single-layer bi-directional RNN as encoder and a single-layer uni-directional RNN as decoder.

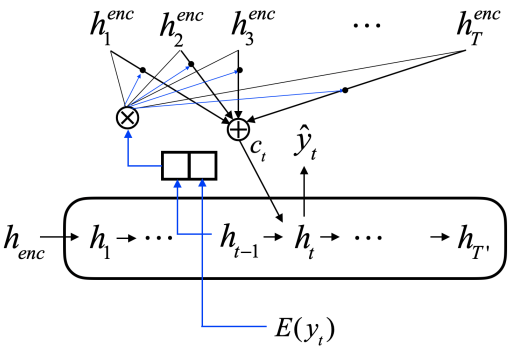

Figure 3: Decoder computation at time t of `RNNattn`.

The initial hidden vector of the decoder is generated through a non-linear transformation on the last hidden vector of the encoder (cf. Cho et al. (2014)):

$$h_0 = \tanh(V h_{enc})$$

2.2 RNN encoder-decoder with attention

The `RNNattn` model is an extension of the `RNNbase` model, with the addition of the attention mechanism during the decoding stage. The attention layer generates a weight for each hidden state in the encoder, which evaluates how important it is for the next token to be translated. Therefore, at each time step, the context vector c_t is a weighted sum of the encoder hidden vectors by attention weights, as illustrated in Figure 3.

2.3 CNN-RNN encoder-decoder with attention

Finally, we introduce our third model: `CNNattn`. In this model, the encoder is replaced a two-layer convolutional encoder (Kalchbrenner et al., 2014). In order to generate the context vector at the encoding stage, a non-linear transformation is applied on the encoder hidden vectors, followed by a fully-connected two-layer feed-forward network. For the non-linear transformation, we choose to use the hyperbolic tangent after experimenting with max-pooling and rectified linear units as well. Otherwise, we employ the same decoder that is used in the `RNNattn` model (cf. Figure 3).

3 Experiment settings

3.1 Dataset

Our dataset consists of the pre-tokenized training, validation, and test splits for the two language pairs:

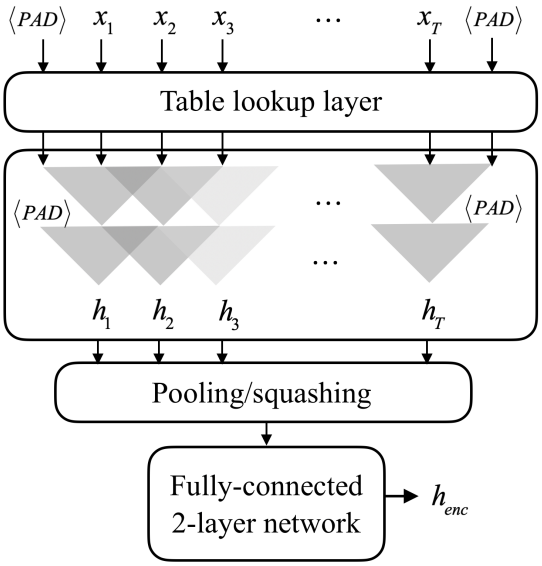

Figure 4: Encoder structure of `CNNattn`.

Vietnamese-English (Vi-En) and Chinese-English (Zh-En). The data was adapted from the original IWSLT 2012 and IWSLT 2013 workshop data.

In total, the Vi-En training split contains 133,317 sentences. The Zh-En split is 1.6 times larger and contains 213,377 training sentences. Figure 5 shows the sentence length distribution for the Vietnamese and Chinese training data. Based on the plots, we remove empty sentences during training and set the sentence truncation length to 80 tokens to speed up the computation. All experiments are executed on the NYU high performance computing (HPC) Prince cluster.

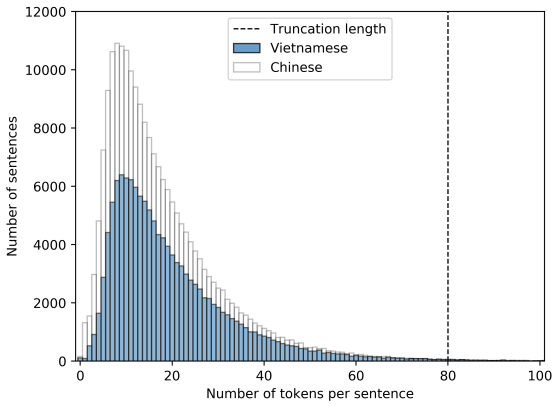

Figure 5: Distribution of number of tokens per sentence.

Hyperparameter	Default setting
Vocabulary	
Vocabulary size	25,000
Use `fastText` embeddings	False
Freeze `fastText` embeddings	False
Model	
Embedding size	300
Hidden size	1,000
Number of layers (encoder)	1
Number of layers (decoder)	1
Number of directions (encoder)	1
Number of directions (decoder)	2
Kernel size (CNN)	3
Maximum sentence length	80
Training	
Teacher forcing ratio	1.0
Beam search width	3
Number of epochs	25
Criterion	`NLLLoss`
Optimizer	`Adam`
Learning rate (encoder)	0.001
Learning rate (decoder)	0.001
Learning rate scheduler	`ExponentialLR`
Gamma	0.95
Learning rate annealing	True
Early stop	True
Batch size	64

Table 1: Default hyperparameter settings for all models.

3.2 Model parameters

The default hyperparameters for all models are listed in Table 1. In addition, we perform an ablation study on the following parameters for each of the three models:

1. Vocabulary size (25,000; 50,000; 100,000) and using 300-dimensional, pre-trained `fastText` word embeddings (Bojanowski et al., 2016)

2. Hidden size (500; 1,000; 2,000) combined with the corresponding learning rate (0.002; 0.001, 0.0005)

3.3 Scoring metric

Our models are evaluated based on their respective bilingual evaluation understudy (BLEU) scores (Papineni et al., 2002). BLEU is a common choice for automatic MT evaluation because it is language-independent, inexpensive, and efficient to compute. The main idea behind the BLEU method is to use a weighted average of variable length phrases that are matched against one or more reference translations.

BLEU scores typically range from 0 to 100, where a score of 100 indicates a generated translation that is identical to a reference translation. The reported BLEU scores are calculated over the entire test set to provide a reasonable estimate of the overall quality of the output translations. Specifically, we use the `SacreBLEU` Python module to compute the corpus BLEU scores, which uses the standard four-gram methodology (Post, 2018).

3.4 Search algorithms

In NMT, there are two widely adopted strategies to generate new translations that approximately maximize the trained conditional probability: greedy search and beam search.

Greedy search is a simple approximation that selects the most likely token at each step in the output sequence. Beam search keeps a set of top hypotheses for each step of the output sequence and explores each of those hypotheses to determine the overall most likely sequence. While greedy search is more straightforward to implement, the resulting output sequences are usually sub-optimal. Selecting the top candidate at each time step effectively discards runner-up hypotheses that might end up more probable when computed over the entire sequence.

Therefore, we also implement beam search. The number of active candidates at any given step, also called the beam width, is a hyperparameter that is empirically evaluated on the validation set. In addition, we experiment with different methods of the length penalty for the sentence score. The basic methodology is to divide the log probability of each resulting sentence by the length of the sentence; a more advanced approach is to divide by a non-linear function of length as described in Wu et al. (2016).

4 Results

4.1 Ablation study

We conduct a grid search over two pairs of parameters: vocabulary size and use of word embeddings, as well as hidden size and learning rate. A subset of the grid search results on the validation set are shown in Table 2. The best parameter settings for each model and language are stated in Table 3.

Overall, we find that a hidden size of 2,000 with a learning rate of 0.0005 works well for all models.

Hyperparameter setting	Vi-En	Zh-En
`RNNbase`		
Vocabulary size = 25,000	14.79	**17.78**
Vocabulary size = 50,000	**19.06**	17.42
Vocabulary size = 100,000	17.27	16.73
Hidden size = 500; LR = 0.002	14.24	15.52
Hidden size = 1,000; LR = 0.001	20.22	16.27
Hidden size = 2,000; LR = 0.0005	**21.67**	**17.83**
`RNNattn`		
Vocabulary size = 25,000	**25.50**	15.01
Vocabulary size = 50,000	23.02	15.10
Vocabulary size = 100,000	19.80	**15.97**
Hidden size = 500; LR = 0.002	19.38	17.02
Hidden size = 1,000; LR = 0.001	25.09	17.57
Hidden size = 2,000; LR = 0.0005	**26.03**	**18.57**
`CNNattn`		
Vocabulary size = 25,000	**21.22**	15.27
Vocabulary size = 50,000	16.36	**17.28**
Vocabulary size = 100,000	14.96	16.98
Hidden size = 500; LR = = 0.002	21.06	14.92
Hidden size = 1,000; LR = = 0.001	**24.09**	16.87
Hidden size = 2,000; LR = = 0.0005	23.57	**17.59**

Table 2: Results of the ablation study per model.

Integrating pre-trained `fastText` embeddings generally does not improve model accuracy, with one exception: the `CNNattn` model for Chinese-English does benefit from the addition of pre-trained word embeddings.

Please note: For the remainder of this paper, all results are based on the fine-tuned model parameter settings in Table 3.

4.2 BLEU scores

Based on the results on the validation set, we first determine the optimal beam width to be 5 for all of our models. Using beam search of width 5 and the best models after tuning hyperparameters, the best final BLEU scores on the Vietnamese and Chinese test splits are stated in Table 4. A number of selected example translations are provided in Table 5.

5 Analysis

We observe several phenomena during the model selection and hyperparameter optimization that have significant impact on the final results. We document them in this section and propose our hypotheses for their cause and effect.

First, we notice a tendency for repeated n-grams in the predictions for all three models. Below is one sample sentence from the English training set

Hyperparameter	Vi-En	Zh-En
`RNNbase`		
Vocabulary size	25,000	25,000
Use `fastText` embeddings	False	False
Hidden size	2,000	2,000
Learning rate (encoder)	0.0005	0.0005
Learning rate (decoder)	0.0005	0.0005
`RNNattn`		
Vocabulary size	25,000	25,000
Use `fastText` embeddings	False	False
Hidden size	2,000	2,000
Learning rate (encoder)	0.0005	0.0005
Learning rate (decoder)	0.0005	0.0005
`CNNattn`		
Vocabulary size	25,000	25,000
Use `fastText` embeddings	False	**True**
Hidden size	2,000	2,000
Learning rate (encoder)	0.0005	0.0005
Learning rate (decoder)	0.0005	0.0005

Table 3: Fine-tuned hyperparameter settings.

Model	BLEU (Vi-En)	BLEU (Zh-En)
`RNNbase`	22.02	20.90
`RNNattn`	**25.37**	20.79
`CNNattn`	25.02	**22.10**

Table 4: Final BLEU scores on the test set.

(**T**) and the corresponding output translation that the `RNNattn` model produces (**P**).

T: *in India and Nepal , I was introduced to the brick*
P: *in India and India , I was sent out to India , I was sent out to the Indian , and I was in the Indian book , I was in the Indian . . .*

We theorize that this behaviour is caused by n-grams that connect disproportionately frequently to their own starting tokens within the same language. In the example above, comma-delimited lists of phrases starting with "I was" appear several times in the training data, which might cause the decoder to enter an infinite loop. In order to overcome this issue, we could potentially introduce a penalty for sub-sequences that have already been predicted during the decoding phase.

Second, we notice that larger hidden sizes in the context vector (up to a size of 2,000) consistently score higher across all models and language pairs. This is likely due to the models' increased capacity to store context information in the added dimensions of the hidden context vector. However, this relationship does not seem to be linear in hidden size, as certain increases in the hidden size improve the predictive accuracy of the model by a greater margin than others. For instance, increasing the hidden size of the Vi-En `RNNattn` model from 500 to 1,000 improves the BLEU validation score by almost 5 points, whereas further increasing the hidden size from 1,000 to 2,000 only yields one additional point of accuracy. Our hypothesis is that for each language, there exists a minimum capacity required to represent the bulk "meaning" of the language; afterwards, the effect of larger hidden vectors abates.

Using pre-trained `fastText` word embeddings yields mixed results. Theoretically, pre-trained word embeddings should help the models converge to the optimal word embeddings faster (Qi et al., 2018). In practice, integrating existing word embeddings only improves model accuracy in a few specific settings. It could be that the general-domain of the `fastText` embeddings is simply not well-aligned with the domain of the given dataset. However, in the case of the `CNNattn` models, the use of pre-trained word embeddings does increase the model accuracy significantly, from 17.59 to 19.87 BLEU score on the Chinese validation set.

With regard to the search algorithm, we observe that using beam search over greedy search consistently improves the validation BLEU scores by a margin of 0-2 points. Ultimately, the best results on the validation set are achieved by setting the beam width to five. Integrating the specialized length penalty factor as described in Wu et al. (2016) does not outperform the standard beam search, although the resulting BLEU scores are still generally higher than those based on greedy search.

Finally, we adopt a canonical method of halving the learning rate after not seeing improvements for 20 steps during training (Goodfellow et al., 2016). Using this technique, we are able to improve model performance by approx. 1-2 points of BLEU score for all of our models. We argue that this method is successful because the global minimum is a very narrow region within the solution space that is easily overshot when the learning rate is too big. Using a larger learning rate in the beginning however is still necessary to sufficiently progress the model training up to a certain point.

Truth	Translation
can we move that up a little bit ?	can we move that up a little bit ?
i wanted them to know that we will be bearing witness to them , and that we will do whatever we can to help make a difference in their lives .	i want them to know that we're going to make them , and we're going to be able to let them move their lives .
for the last 28 years , I've been documenting indigenous cultures in more than 70 countries on six continents , and in 2009 I had the great honor of being the sole <UNK> at the Vancouver Peace Summit .	over the last 50 years , I've been documenting over the global culture of about 100 countries on the continent , and so I got my right to become the only black marriage in the town of the World War .

Table 5: Sample translations from the Vi-En training set.

6 Summary and conclusion

In this paper, we introduce three model variants for Vietnamese-English and Chinese-English machine translation: `RNNbase`, `RNNattn`, and `CNNattn`. Overall, we confirm that the use of the attention mechanism consistently and significantly improves model performance. In addition, evaluating the trained models using beam search instead of greedy search yields a few extra points of accuracy.

In general, our best model for the Vietnamese-English dataset is the `RNNattn` model, with a BLEU score of 25.37 on the Vietnamese test split. For Chinese-English, our best model is the `CNNattn` model, which achieves a BLEU score of 22.10 on the Chinese test set.

In the future, we will experiment with additional hyperparameter optimization techniques, such as varying the kernel size for the `CNNattn` model, or employing the self-attention mechanism to exploit connections between tokens in the same sequence. A larger and more diverse dataset with enhanced preprocessing would likely improve the predictive accuracy of our best models as well.

References

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

[Bojanowski et al.2016] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

[Brown et al.1990] Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. 1990. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

[Cho et al.2014] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

[Goodfellow et al.2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[Kalchbrenner et al.2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188.

[Koehn et al.2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Nallapati et al.2016] Ramesh Nallapati, Bing Xiang, and Bowen Zhou. 2016. Sequence-to-Sequence RNNs for Text Summarization. *CoRR*, abs/1602.06023.

[Papineni et al.2002] 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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Post2018] Matt Post. 2018. A call for clarity in reporting BLEU scores. *CoRR*, abs/1804.08771.

[Qi et al.2018] Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? *CoRR*, abs/1804.06323.

[Venugopalan et al.2015] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney,

Trevor Darrell, and Kate Saenko. 2015. Sequence to sequence - video to text. In *The IEEE International Conference on Computer Vision (ICCV)*, December.

[Vinyals and Le2015] Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *CoRR*, abs/1506.05869.

[Williams et al.2016] Philip Williams, Rico Sennrich, Matt Post, and Philipp Koehn. 2016. Syntax-based statistical machine translation. *Synthesis Lectures on Human Language Technologies*, 9(4):1–208.

[Wu et al.2016] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144.

[Zaremba and Sutskever2014] Wojciech Zaremba and Ilya Sutskever. 2014. Learning to Execute. *CoRR*, abs/1410.4615.