Variable-Length Word Encodings for Neural Translation Models

Rohan Chitnis and **John DeNero**
Computer Science Division
University of California, Berkeley
{ronuchit,denero}@berkeley.edu

Abstract

Recent work in neural machine translation has shown promising performance, but the most effective architectures do not scale naturally to large vocabulary sizes. We propose and compare three variable-length encoding schemes that represent a large vocabulary corpus using a much smaller vocabulary with no loss in information. Common words are unaffected by our encoding, but rare words are encoded using a sequence of two pseudo-words. Our method is simple and effective: it requires no complete dictionaries, learning procedures, increased training time, changes to the model, or new parameters. Compared to a baseline that replaces all rare words with an *unknown word* symbol, our best variable-length encoding strategy improves WMT English-French translation performance by up to 1.7 BLEU.

1 Introduction

Bahdanau et al. (2014) propose a neural translation model that learns vector representations for individual words as well as word sequences. Their approach jointly predicts a translation and a latent word-level alignment for a sequence of source words. However, the architecture of the network does not scale naturally to large vocabularies (Jean et al., 2014).

In this paper, we propose a novel approach to circumvent the large-vocabulary challenge by preprocessing the source and target word sequences, encoding them as a longer token sequence drawn from a small vocabulary that does not discard any information. Common words are unaffected, but rare words are encoded as a sequence of two pseudo-words. The exact same learning and inference machinery applied to these transformed data yields improved translations.

We evaluate a family of 3 different encoding schemes based on Huffman codes. All of them eliminate the need to replace rare words with the *unknown word* symbol. Our approach is simpler than other methods recently proposed to address the same issue. It does not introduce new parameters into the model, change the model structure, affect inference, require access to a complete dictionary, or require any additional learning procedures. Nonetheless, compared to a baseline system that replaces all rare words with an *unknown word* symbol, our encoding approach improves English-French news translation by up to 1.7 BLEU.

2 Background

2.1 Neural Machine Translation

Neural machine translation describes approaches to machine translation that learn from corpora in a single integrated model that embeds words and sentences into a vector space (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014). We focus on one recent approach to neural machine translation, proposed by Bahdanau et al. (2014), that predicts both a translation and its alignment to the source sentence, though our technique is relevant to related approaches as well.

The architecture consists of an encoder and a decoder. The encoder receives a source sentence $\mathbf{x}$ and encodes each prefix using a recurrent neural network that recursively combines embeddings x_j for each word position j:

$$\overrightarrow{h}_j = f(x_j, \overrightarrow{h}_{j-1}) \quad (1)$$

where f is a non-linear function. Reverse encodings $\overleftarrow{h}_j$ are computed similarly to represent suffixes of the sentence. These vector representations are stacked to form h_j, a representation of the

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 2088–2093,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

whole sentence focused on position j.

The decoder predicts each target word y_i sequentially according to the distribution

$$P(y_i|y_{i-1}, ..., y_1, \mathbf{x}) = g(y_{i-1}, s_i, c_i) \quad (2)$$

where s_i is a hidden decoder state summarizing the prefix of the translation generated so far, c_i is a summary of the entire input sequence, and g is another non-linear function. Encoder and decoder parameters are jointly optimized to maximize the log-likelihood of a training corpus.

Depending on the approach to neural translation, c can take multiple forms. Bahdanau et al. (2014) propose integrating an attention mechanism in the decoder, which is trained to determine on which portions of the source sentence to focus. The decoder computes c_i, the summarizing context vector, as a convex combination of the h_j. The coefficients of this combination are proportional (softmax) to an alignment model prediction $\exp a(h_j, s_i)$, where a is a non-linear function.

The speed of prediction scales with the output vocabulary size, due to the denominator of Equation 2 (Jean et al., 2014). The input vocabulary size is also a challenge for storage and learning. As a result, neural machine translation systems only consider the top 30K to 100K most frequent words in a training corpus, replacing the other words with an *unknown word* symbol.

2.2 Related Work

There has been much recent work in improving translation quality by addressing these vocabulary size challenges. Luong et al. (2014) describe an approach that, similar to ours, treats the translation system as a black box. They eliminate unknown symbols by training the system to recognize from where in the source text each unknown word in the target text came, so that in a postprocessing phase, the unknown word can be replaced by a dictionary lookup of the corresponding source word. In contrast, our method does not rely on access to a complete dictionary, and instead transforms the data to allow the system itself to learn translations for even the rare words.

Some approaches have altered the model to circumvent the expensive normalization computation, rather than applying preprocessing and postprocessing on the text. Jean et al. (2014) develop an importance sampling strategy for approximating the softmax computation. Mnih and Kavukcuoglu (2013) present a technique for approximation of the target word probability using noise-contrastive estimation.

Sequential or hierarchical encodings of large vocabularies have played an important role in recurrent neural network language models, primarily to address the inference time issue of large vocabularies. Mikolov et al. (2011b) describe an architecture in which output word types are grouped into classes by frequency: the network first predicts a class, then a word in that class. Mikolov et al. (2013) describe an encoding of the output vocabulary as a binary tree. To our knowledge, hierarchical encodings have not been applied to the input vocabulary of a machine translation system.

Other methods have also been developed to work around large-vocabulary issues in language modeling. Morin and Bengio (2005), Mnih and Hinton (2009), and Mikolov et al. (2011a) develop hierarchical versions of the softmax computation; Huang et al. (2012) and Collobert and Weston (2008) remove the need for normalization, thus avoiding computation of the summation term over the entire vocabulary.

2.3 Huffman Codes

An encoding can be used to represent a sequence of tokens from a large vocabulary $\mathcal{V}$ using a small vocabulary $\mathcal{W}$. In the case of translation, let $\mathcal{V}$ be the original corpus vocabulary, which can number in the millions of word types in a typical corpus. Let $\mathcal{W}$ be the vocabulary size of a neural translation model, typically set to a much smaller number such as 30,000.

A deterministically invertible, variable-length encoding maps each $v \in \mathcal{V}$ to a sequence $w \in \mathcal{W}+$ such that no other $v' \in \mathcal{V}$ is mapped to a prefix of w. Encoding simply replaces each element of $\mathcal{V}$ according to the map, and decoding is unambiguous because of this prefix restriction. An encoding can be represented as a tree in which each leaf corresponds to an element of $\mathcal{V}$, each node contains a symbol from $\mathcal{W}$, and the encoding of any leaf is its path from the root.

A Huffman code is an optimal encoding that uses as few symbols from $\mathcal{W}$ as possible to encode an original sequence of symbols from $\mathcal{V}$. Although binary codes are typical, $\mathcal{W}$ can have any size. An optimal encoding can be found using a greedy algorithm (Huffman, 1952).

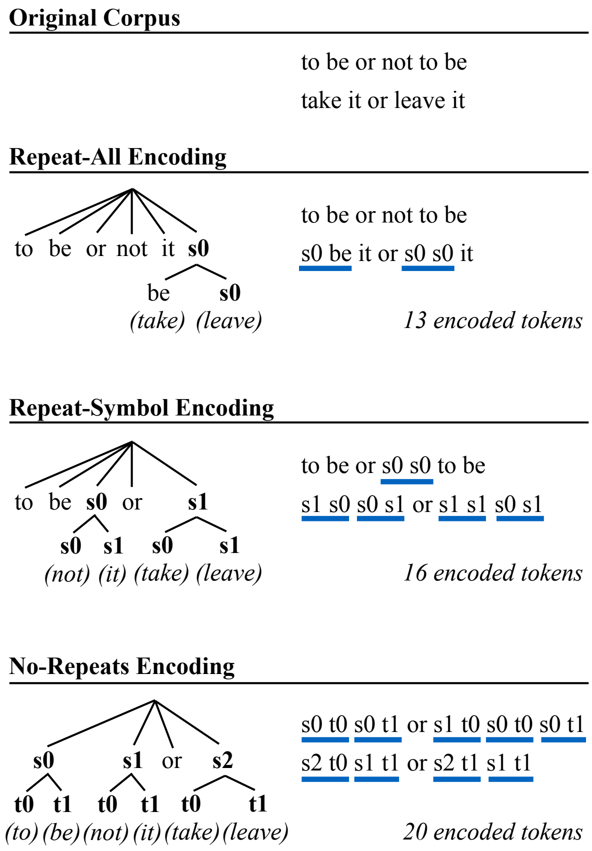

Figure 1: Our three encoding schemes are applied to a two-sentence toy corpus for which each word type appears one or two times, and the total vocabulary size V is 7. An optimal encoding tree under each scheme is shown for an encoded vocabulary size W of 6. As stricter constraints are imposed on the encoding, the encoded corpus length increases and the number of elements of V that can be represented using a single symbol decreases. Two-symbol encodings of rare words are underlined.

3 Variable-Length Encoding Methods

We consider three different encoding schemes that are based on Huffman codes. The encoding for a toy corpus under each scheme is depicted in Figure 1. While a Huffman code achieves the shortest possible encoded length using a fixed vocabulary size W, symbols are often shared between both common words and rare words. The variants we consider are designed to prevent specific forms of symbol sharing across encodings.

3.1 Encoding Schemes

Repeat-All. The first scheme is a standard Huffman code. In our experiments with $V \approx 2 \cdot 10^6$, $W = 3 \cdot 10^4$, and frequencies drawn from the WMT corpus, all words in $\mathcal{V}$ are encoded as either a single symbol or two symbols of $\mathcal{W}$. We denote the single-symbol words (which have the highest frequency) as *common*, and we call the other words *rare*. The *Repeat-All* encoding scheme has the highest number of common words. In Figure 1, common words are represented as themselves. Rare words are represented by two words, and the first is always a pseudo-word symbol introduced into $\mathcal{W}$ of the form **sX** for an integer X.

Repeat-Symbol. The *Repeat-Symbol* encoding scheme does not allow common-word symbols to appear in the encoding of rare words. Instead, each rare word is encoded as a two-symbol sequence of the form "**sX sY**," where X and Y are integers that may be the same or different. This scheme decreases the number of common words in order to encode all rare words using a restricted set of symbols. In this scheme, a common word in the encoded vocabulary always corresponds to a common word in the original vocabulary, reducing ambiguity of common word symbols at the expense of increasing ambiguity of pseudo-word symbols.

No-Repeats. Our final encoding scheme, *No-Repeats*, uses a different vocabulary for the first and second symbols in each rare word. That is, rare words are represented as "**sX tY**," where X and Y are integers that may be the same or different. In this scheme, common words and rare words do not share symbols, and each symbol can immediately be identified as common, the first of a rare encoding pair, or the second of a rare encoding pair.

3.2 Symbol Counts

To maximize performance, it is critical to set the number of common words (which transform to themselves) as high as possible while satisfying the desired total vocabulary size, counting all the newly introduced symbols. In this section, we algebraically derive this optimal number of common words for each encoding scheme. We define the following:

V: Size of the original vocabulary.

W: Size of the encoded vocabulary.

C: Number of common words.

S: Number of pseudo-words of the form **sX**.

T: Number of pseudo-words of the form **tX**.

We are interested in maximizing C so that total encoding length is minimized.

Repeat-All. We would like to encode the $V - C$ rare words, using only $W - C$ new symbols. To do so, for each new symbol (non-terminal node in our encoding tree), we have all W symbols under it in that branch. Therefore, we maximize C satisfying the constraint that

$$V - C \leq (W - C) \cdot W$$

Repeat-Symbol. Out of the $V - C$ rare words, we would like to pack them into a complete tree so that they may be encoded using our remaining $W - C$ symbols. Therefore, we maximize C satisfying the constraint that

$$V - C \leq (W - C)^2$$

No-Repeats. Again, we desire to pack $V - C$ rare words into a complete tree where we may use $W - C$ symbols. To maximize C, we let $S = T$. Because $S + T + C = W$, we have that $2S + C = W$. Therefore, we maximize C satisfying the constraint that

$$V - C \leq \left(\frac{W - C}{2}\right)^2$$

4 Experimental Results

We trained a public implementation[1] of the system described in Bahdanau et al. (2014) on the English-French parallel corpus from ACL WMT 2014, which contains 348M tokens. We evaluated on news-test-2014, also from WMT 2014, which contains 3003 sentences. All experiments used the same learning parameters and vocabulary size of 30,000.

We constructed each encoding by the following method. First, we used the formulas derived in the previous section to calculate the optimal number of common words C for each encoding scheme, using V to be the true vocabulary size of the training corpus and $W = 30,000$. We then found the C most common words in the text and encoded them as themselves. For the remaining rare words, we encoded them using a distinct symbol whose form matched the one prescribed for each encoding scheme. The encoding was then applied separately to both the source text and the target text. Our encoding schemes all increased the total number of tokens in the training corpus by approximately 4%.

Encoding	BLEU	# Common Words
None	25.77	30,000
Repeat-All	27.45	29,940
Repeat-Symbol	26.52	28,860
No-Repeats	25.79	27,320

Table 1: BLEU scores (%) on detokenized test set for each encoding scheme after training for 5 days.

To construct the mapping from rare words to their 2-word encodings, we binned rare words by frequency into branches. Thus, rare words of similar frequency in the training corpus tended to have encodings with the same first symbol. Similarly, the standard Huffman construction algorithm groups together rare words with similar frequencies within subtrees. More intelligent heuristics for constructing trees, such as using translation statistics instead of training corpus frequency, would be an interesting area of future work.

4.1 Results

We used the RNNsearch-50 architecture from Bahdanau et al. (2014) as our machine translation system. We report results for this system alone, as well as for each of our three encoding schemes, using the BLEU metric (Papineni et al., 2002). Table 1 summarizes our results after training each variant for 5 days, corresponding to roughly 2 passes through the 180K-sentence training corpus.

Alternative techniques that leverage bilingual resources have been shown to provide larger improvements. Jean et al. (2014) demonstrate an improvement of 3.1 BLEU by using bilingual word co-occurrence statistics in an aligned corpus to replace *unknown word* tokens. Luong et al. (2014) demonstrate an improvement of up to 2.8 BLEU over a series of stronger baselines using an unknown word model that also makes predictions using a bilingual dictionary.

4.2 Analysis

Our results indicate that the encoding scheme that keeps the highest number of common words, *Repeat-All*, performs best. Table 2 shows the unigram precision of each output. The common word translation accuracy is higher for all encoding schemes than for the baseline, although all preci-

[1] `github.com/lisa-groundhog/GroundHog`

Encoding	Common	Rare	1st Symbol
None	62.0	0.0	-
Repeat-All	65.8	28.0	64.8
Repeat-Symbol	65.5	16.5	24.8
No-Repeats	63.6	15.8	25.7

Table 2: Test set precision (%) on common words and rare words for each encoding strategy. *1st Symbol* denotes the precision of the first pseudo-word symbol in an encoded rare word.

sions are similar. Larger differences appear in the precision of rare words. The scheme that encodes rare words using both pseudo-words and common words gives substantially higher rare word accuracy than any other approach.

The final column of Table 2 shows the unigram precision of the first pseudo-word in an encoded rare word. The *Repeat-All* scheme uses only 60 different first symbols to encode all rare words. The other schemes require over 1,000. The fact that *Repeat-All* has a constrained set of rare word first symbols may account for its higher rare word precision.

It is possible for the model to predict an invalid encoded sequence that does not correspond to any word in the original vocabulary. However, in our experiments, we did not observe any such sequences in the decoding of the test set. A reasonable way to deal with invalid sequences would be to drop them from the output during decoding.

5 Conclusion and Future Work

We described a novel approach for encoding the source and target text based on Huffman coding schemes, eliminating the use of the *unknown word* symbol. An important continuation of our work would be to develop heuristics for effectively grouping "similar" words in the source and target text, so that they tend to have encodings that share a symbol. Even with our naive grouping by corpus frequency, our approach offers a simple way to predict both common and rare words in a neural translation model. As a result, performance improves by up to 1.7 BLEU. We expect that the simplicity of our technique will allow for straightforward combination with other enhancements and neural models.

References

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

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.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning*.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the Association for Computational Linguistics*.

David A. Huffman. 1952. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *CoRR*, abs/1412.2007.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models.

Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *CoRR*, abs/1410.8206.

Tomáš Mikolov, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černocký. 2011a. Strategies for training large scale neural network language models. In *Proceedings of ASRU*.

Tomáš Mikolov, S. Kombrink, L. Burget, J.H. Cernocky, and Sanjeev Khudanpur. 2011b. Extensions of recurrent neural network language model. In *Acoustics, Speech and Signal Processing*.

Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Andriy Mnih and Geoffrey E. Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems*.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *AI Stats*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.