MetaMind Neural Machine Translation System for WMT 2016

James Bradbury
MetaMind - A Salesforce Company
Palo Alto, CA
james.bradbury@salesforce.com

Richard Socher
MetaMind - A Salesforce Company
Palo Alto, CA
rsocher@salesforce.com

Abstract

Neural Machine Translation (NMT) systems, introduced only in 2013, have achieved state of the art results in many MT tasks. MetaMind's submissions to WMT '16 seek to push the state of the art in one such task, English→German news-domain translation. We integrate promising recent developments in NMT, including subword splitting and back-translation for monolingual data augmentation, and introduce the Y-LSTM, a novel neural translation architecture.

1 Introduction

The field of Neural Machine Translation (NMT), which seeks to use end-to-end neural networks to translate natural language text, has existed for only three years. In that time, researchers have explored architectures ranging from convolutional neural networks (Kalchbrenner and Blunsom, 2013) to recurrent neural networks (Chung et al., 2014) to attentional models (Bahdanau et al., 2015; Luong et al., 2015) and achieved better performance than traditional statistical or syntax-based MT techniques on many language pairs. NMT models first achieved state-of-the-art performance on the WMT English→German news-domain task in 2015 (Luong et al., 2015) and subsequent improvements have been reported since then (Sennrich et al., 2015a; Li and Jurafsky, 2016).

The problem of machine translation is fundamentally a sequence-to-sequence transduction task, and most approaches have been based on an encoder-decoder architecture (Sutskever et al., 2014; Cho et al., 2014). This entails coupled neural networks that encode the input sentence into a vector or set of vectors and decode that vector representation into an output sentence in a different language respectively. Recently, a third component has been added to many of these models: an attention mechanism, whereby the decoder can attend directly to localized information from the input sentence during the output generation process (Bahdanau et al., 2015; Luong et al., 2015). The encoder and decoder in these models typically consist of one-layer (Cho et al., 2014) or multi-layer recurrent neural networks (RNNs); we use four- and five-layer long short-term memory (LSTM) RNNs. The attention mechanism in our four-layer model is what Luong (2015) describes as "Global attention (dot)"; the mechanism in our five-layer Y-LSTM model is described in Section 2.1.

Every NMT system must contend with the problem of unbounded output vocabulary: systems that restrict possible output words to the most common 50,000 or 100,000 that can fit comfortably in a softmax classifier will perform poorly due to large numbers of "out-of-vocabulary" or "unknown" outputs. Even models that can produce every word found in the training corpus for the target language (Jean et al., 2015) may be unable to output words found only in the test corpus. There are three main techniques for achieving fully open-ended decoder output. Models may use computed alignments between source and target sentences to directly copy or transform a word from the input sentence whose corresponding translation is not present in the vocabulary (Luong et al., 2015) or they may conduct sentence tokenization at the level of individual characters (Ling et al., 2015) or subword units such as morphemes (Sennrich et al., 2015b). The latter techniques allow the decoder to construct words it has not previously encountered out of known characters or morphemes; we apply the subword splitting strategy using Morfessor 2.0, an unsupervised morpheme segmentation model (Virpioja et

Proceedings of the First Conference on Machine Translation, Volume 2: Shared Task Papers, pages 264–267,
Berlin, Germany, August 11-12, 2016.

al., 2013).

Another focus of recent research has been ways of using monolingual corpus data, available in much larger quantities, to augment the limited parallel corpora used to train translation models. One way to accomplish this is to train a separate monolingual language model on a large corpus of the target language, then use this language model as an additional input to the decoder or for re-ranking output translations (Gülçehre et al., 2015). More recently, Sennrich (2015b) introduced the concept of augmentation through back-translation, where an entirely separate translation model is trained on a parallel corpus from the target language to the source language. This backwards translation model is then used to machine-translate a monolingual corpus from the target language into the source language, producing a pseudo-parallel corpus to augment the original parallel training corpus. We extend this back-translation method by translating a very large monolingual German corpus into English, then concatenating a unique subset of this augmentation corpus to the original parallel corpus for each training epoch.

2 Model Description

The model identified as `metamind-single` is based on the attention-based encoder-decoder framework described in Luong (2015), using the attention mechanism referred to as "Global attention (dot)." The encoder is a four-layer stacked LSTM recurrent neural network whose inputs (at the bottom layer) are vectors w_t^{in} corresponding to the subword units in the input sentence and which saves the topmost output state at each timestep e_t as the variable-length encoding matrix E. The decoder also contains a four-layer stacked LSTM whose states (c_0 and h_0 for each layer) are initialized to the last states for each layer of the encoder. At the first timestep, the decoder LSTM receives as input an initialization word vector w_0^{out}; its topmost output state h_t is concatenated with an encoder context vector κ_t computed as:

$$\begin{aligned}\text{score}(h_t, e_s) &= h_t e_s^\top \\ \alpha_{st} &= \text{softmax}_{\text{all } s}(\text{score}(h_t, e_s)) \\ \kappa_t &= \sum_s \alpha_{st} e_s\end{aligned}$$

This concatenated output is then fed through an additional neural network layer to produce a final attentional output vector $\tilde{h}$, which serves as input to the output softmax:

$$\begin{aligned}\tilde{h} &= \tanh(W_{\text{att}}[h_t; \kappa_t]) \\ \text{output probabilities} &= \text{softmax}(W_{\text{out}}\tilde{h})\end{aligned}$$

For subsequent timesteps, the decoder LSTM receives as input the previous word vector w_{t-1}^{out} concatenated with the previous output vector $\tilde{h}$.

Decoding is performed using beam search, with beam width 16. The beam search decoder differs slightly from Luong (2015) in that we normalize output sentence probabilities by length, following Cho (2014), rather than performing ad-hoc adjustments to correct for short output sentences.

2.1 Y-LSTM Model

The model identified as `metamind-ylstm` uses a novel attentional framework we call the Y-LSTM. The encoder is a five-layer stacked LSTM recurrent neural network language model (RNN-LM) with subword-vector inputs w_t^{in}, whose topmost output state h_t^{top} is used as input to a softmax layer which predicts the next input token. The middle ($l = 3$) layer of this encoder RNN-LM is connected recurrently to a single-layer LSTM called the "tracker;" a_t denotes the set of inputs to a given LSTM layer:

$$\begin{aligned}a_t^{l\neq 3} &= [h_t^{l-1}; h_{t-1}^l] \\ a_t^{l=3} &= [h_t^{l-1}; h_{t-1}^l; h_{t-1}^{\text{tracker}}] \\ a_t^{\text{tracker}} &= [h_{t-1}^{\text{tracker}}; h_t^3]\end{aligned}$$

The hidden and memory states c_t^{tracker} and h_t^{tracker} of the tracker LSTM are saved at each timestep as the variable-length encoding matrices C and H. The decoder is an analogous RNN-LM with a tracker LSTM, identical except that the hidden and memory states of the decoder's tracker ($\tilde{c}_t^{\text{tracker}}$ and $\tilde{h}_t^{\text{tracker}}$) are replaced at each timestep with an attentional sum of the encoder's saved tracker states:

$$\begin{aligned}\text{score}(\tilde{h}_t, h_s) &= \tilde{h}_t h_s^\top \\ \alpha_{st} &= \text{softmax}_{\text{all } s}(\text{score}(\tilde{h}_t^{\text{tracker}}, h_s^{\text{tracker}})) \\ \tilde{c}_t^{\text{tracker}} &= \sum_s \alpha_{st} c_s^{\text{tracker}} \\ \tilde{h}_t^{\text{tracker}} &= \sum_s \alpha_{st} h_s^{\text{tracker}}\end{aligned}$$

System	BLEU-c on `newstest2016`
Best phrase-based system (`uedin-syntax`)	30.6
Other NMT systems – single model	
NYU/U. Montreal character-based	30.8
U. Edinburgh subword-based (`uedin-nmt-single`)	31.6
Other NMT systems – ensemble or model combination	
U. Edinburgh ensemble of 4 (`uedin-nmt-ensemble`)	34.2
Our systems – single model	
`metamind-single`	31.6
`metamind-ylstm`	29.3
Our systems – ensemble	
`metamind-ensemble`	32.3
Ensemble of four checkpoints without Y-LSTM	32.1

Table 1: BLEU results on the official WMT 2016 test set. Only our main ensemble was entered into the human ranking process, coming in second place behind U. Edinburgh.

The overall network loss is the sum of the language model (negative log-likelihood over the output softmax) losses for the encoder and decoder.

3 Experiment Description

Initial tokenization and preprocessing of the WMT 2016 English→German news translation dataset was performed using the standard scripts provided with Moses (Koehn et al., 2007). Two further processing steps were used to create the subword-based training dataset. First, capitalized characters were replaced with a sequence of a capitalization control character (a Unicode private-use character) and the corresponding lowercase character, in order to allow the subword splitting algorithm to treat capitalized words as either inherently capitalized or capitalized versions of lowercase words. Without this step, much of the limited output softmax capacity is taken up with capitalized variants of common lowercase words; performing this transformation also allows us to forego "truecasing," which removes sentence-initial capitalization in a lossy and sometimes unhelpful way. Second, the capitalization-transformed training corpus for each language is ingested by a Morfessor 2.0 instance configured to use a balance between corpus and vocabulary entropy that produces a vocabulary of approximately 50,000 subword units.

For all experiments, we used using plain stochastic gradient descent with learning rate 0.7, gradient clipping at magnitude 5.0, dropout of 0.2, and learning rate decay of 50% per epoch after 8 epochs.

Following Sennrich (2015b), we first trained a non-Y-LSTM model in the reverse direction (German→English) on the full WMT '16 training corpus (4.4 million sentences). This model was then used simultaneously on 8 GPUs (with a beam search width of 4 for speed purposes) to translate 45 million sentences of the 2014 monolingual German news crawl into English. A full copy of the original training corpus was then concatenated with a unique subset of this augmentation corpus to create a new training corpus for each epoch from 1 to 10; the corpus for epoch 1 was then repeated as epoch 11 *et cetera*.

For `metamind-single`, we trained a non-Y-LSTM model using these augmented corpora, with data-parallel synchronous SGD across four GPUs enabling a batch size of 384 and training speed of about 2,500 subword units per second. The run submitted as `metamind-single` uses a single snapshot of this model after 12 total training epochs.

For `metamind-ylstm`, we trained a Y-LSTM model using the same corpora, with data-parallel synchronous SGD across four GPUs enabling a batch size of 320 and training speed of about 1,500 subword units per second. The run submitted as `metamind-ylstm` uses a single snapshot of this model after 9 total training epochs.

The run submitted as `metamind-ensemble` uses an equally-weighted ensemble of three snapshots of the `metamind-single` model (after 10, 11, and 12 epochs) and a single snapshot of the `metamind-ylstm` model after 9 total training epochs.

4 Results

Results for all three runs described above are presented in Table 1. Only the ensemble was submitted to the human evaluation process, with a final ranking of second place (behind U. Edinburgh's ensemble of four independently initialized models). Our best single model matches the performance of the best model from U. Edinburgh, which applies a similar attentional framework, subword splitting, and back-translated augmentation.

The Y-LSTM model underperformed relative to the model based on Luong (2015), but provided a small additional boost to the ensemble. The primary contribution of this model is to demonstrate that *purely* attentional NMT is possible: the only inputs to the decoder are through the attention mechanism. This may be helpful for using translation to build general attentional sentence encoding models, since the representation of the input sentence is entirely in the attentional encoding, not split between an attentional encoding vector and a vector representing the last timestep of the multilayer encoder hidden state.

Acknowledgements

We would like to thank the developers of Chainer (Tokui et al.,), which we used for all models and experiments reported here. We also thank Stephen Merity, Kai Sheng Tai, and Caiming Xiong for their helpful feedback, and all participants in the manual evaluation campaign. We thank the Salesforce acquisition and IT teams for keeping the MetaMind compute cluster up and running throughout the acquisition process.

References

D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loïc Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *ACL*.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Jiwei Li and Daniel Jurafsky. 2016. Mutual information and diverse decoding improve neural machine translation. *CoRR*, abs/1601.00372.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W. Black. 2015. Character-based neural machine translation. *CoRR*, abs/1511.04586.

M. T. Luong, H. Pham, and C. D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015a. Improving neural machine translation models with monolingual data. *CoRR*, abs/1511.06709.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015b. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

I. Sutskever, O. Vinyals, and Q. V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Seiya Tokui, Kenta Oono, and Shohei Hido. Chainer: a next-generation open source framework for deep learning.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, Mikko Kurimo, et al. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline.