A Survey on Seq2Seq Models for Neural Machine Translation

YU Xinyuan
xyuaj@connect.ust.hk

Abstract

Sequence-to-sequence (seq2seq) model refer to all models that solve tasks by mapping one sequence to another. While seq2seq tasks differ in their nature, the principles are shared behind seq2seq models for different tasks. This survey focuses on seq2seq models for machine translation tasks, in particular neural machine translation (NMT) models. The paper surveys different structure of NMT models, including various encoder and decoder structures and different attention mechanisms. It further investigates different training methods for NMT models in supervised, semi-supervised and unsupervised manners.

1 Introduction

Sequence-to-sequence model refer to all models that map one sequence to another (Neubig, 2017). A broad spectrum of tasks could be viewed as sequence-to-sequence problems, such as dialog systems, speech recognition and machine translation. While these tasks differ in nature, seq2seq models for these tasks share a lot of principles behind. Therefore, this survey focuses on seq2seq models for machine translation tasks, in particular neural machine translation models.

Neural machine translation is one of the most successful application of seq2seq models. It is also one of the driving tasks behind the development of new seq2seq models (Neubig, 2017). NMT models provides a powerful solution to machine translation yet keeps a simple end-to-end structure (Wu et al., 2016; Hassan et al., 2018) compared to traditional phrase based statistical machine translation, which typically consists of several sub-components like language model and translation model.

The basic idea of NMT models consists of an encoder and a decoder. The encoder transforms a sentence $\mathbf{x} = (x_1, x_2, ..., x_{|\mathbf{x}|})$ of source language into sentence representations. Based on the representations, the decoder generates another sentence $\mathbf{y} = (y_1, y_2, ..., y_{|\mathbf{y}|})$ of target language by maximizing the conditional probability of $\mathbf{y}$ given $\mathbf{x}$ i.e. $\arg\max_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x})$.

This survey aims at introducing and comparing different structure of NMT models as well as identifying the contribution and incentives of different training methods in supervised, semi-supervised and unsupervised manners.

2 Model Structures

2.1 RNN Encoder-Decoder

Cho et al. (2014b) and Sutskever et al. (2014) proposed NMT models that used two recurrent neural networks (RNN) for encoding and decoding respectively. Both models shared the same design principle as the encoder encoded the whole source sentence into a fixed-length vector and the decoder RNN generated target words one at a time.

Cho et al. (2014b) adopted one layer RNN with gated recurrent units (GRU) for encoding and decoding. The encoder read words of input sentence $\mathbf{x}$ sequentially and summarized the whole sentence into a context vector c, where c was the final hidden state $h_{|\mathbf{x}|}$ of RNN. The decoder differed from the encoder in that the hidden states and outputs were conditioned on both previous output y_{t-1} and context vector c. The encoder's states h was computed by,

$$h_t = f(h_{t-1}, x_t) \tag{1}$$

and similarly, the decoder's states and outputs were given by,

$$s_t = f(s_{t-1}, y_{t-1}, c) \tag{2}$$

$$P(y_t \mid y_{t-1}, ..., y_1, c) = g(s_t, y_{t-1}, c) \quad (3)$$

where f and g were activation functions. This work successfully explores the effectiveness of NMT models, however the performance of model drops drastically for long sentences (Cho et al., 2014a).

Sutskever et al. (2014) adopted 2-layer RNN with long short-term memory (LSTM) cells for encoding and decoding. The model essentially differed from the model of Cho et al. (2014b) in that the context vector was only used to initialize hidden states of the decoder and was not involved in further computation. Despite the simplicity, Sutskever et al. (2014) managed to handle long sentences by reversing the order of input sentence. In the original paper, it was argued that reversing the order of input sentence would introduce short term dependencies between the input and output words, which would boost the training performance.

2.2 RNN Encoder-Decoder with Attention Mechanism

Ideally, RNN should encode all information for translation into the fixed-length context vector. However, RNN is not good at capturing long-distance dependencies and the models cannot handle long sentences properly (Cho et al., 2014a). This gap between ideals and realities suggests that the fixed-length vector is the bottleneck in improving NMT models' performance.

Inspired by the alignment process of traditional phrase based statistical machine translation, Bahdanau et al. (2014) first proposed to add an attention mechanism (also called global attention (Luong et al., 2015)) to NMT models. The original paper followed up with the encoder-decoder model of Cho et al. (2014b) and replaced the encoder with a bi-directional RNN in addition to adding the attention layer.

The basic idea of attention mechanism (Bahdanau et al., 2014) is that instead of learning a single vector representation for input sentence, the model keeps vector representations of every input word. This is done by concatenating the states of encoder $H = concat(h_1, ..., h_{|\mathbf{x}|})$. The context vector is then computed as weighted sum over all representations

$$c_t = H * \alpha_t \quad (4)$$

at each decoding step, where α_t is the attention vector compute as

$$\alpha_{tj} = \frac{exp(score(s_{t-1}, h_j))}{\sum_k exp(score(s_{t-1}, h_k))} \quad (5)$$

and $score$ is the attention score function to be introduced later. This idea relaxes the requirement of encoding everything into a fixed-length vector. Hence, the decoder gets to decide how much attention to pay for each input word.

The $score$ function in Eq.(5) computes a score of decoder state and encoder representation. Luong et al. (2015) proposed three ways to compute the score:

dot: The simplest way to compute score is by taking dot products of two vectors: $score(s_t, h_j) = s_t^T h_j$. However, this would require the encoder and decoder states to be in the same vector space.

general: the general score function relaxes the constraint of dot product by adding an extra linear transformation: $score(s_t, h_j) = s_t^T W_a h_j$.

concat: The most general way to compute score is to use a neural network with one hidden layer: $score(s_t, h_j) = v_a^T \tanh(W_a[s_t; h_j])$. This method was adopted in the original paper of Bahdanau et al. (2014).

Besides different score functions, Luong et al. (2015) also proposed two attention mechanism, namely global attention and local attention. The global attention model essentially resembled the model of Bahdanau et al. (2014). The local attention model, on the other hand, computed the context vector over a dynamically determined window $[p_t - D, p_t + D]$. The window parameter D was a hyper-parameter and p_t could either be monotonic aligned as $p_t = t$ or be predictive aligned as $p_t =\mid \mathbf{y} \mid \cdot \sigma(v_p^T \tanh(W_p s_t))$. In experiments of Luong et al. (2015), dot score worked well for global attention and general score did better for local attention.

2.3 Encoder-Decoder beyond RNN

Despite the success of attention mechanism, RNN cannot be efficiently paralleled on GPU and often fail to capture long term dependencies (Gehring et al., 2017; Vaswani et al., 2017). Recently proposed NMT models dispensed with recurrent neural networks completely for the sake of parallelism.

ConvS2S

Gehring et al. (2017) proposed a NMT model named ConvS2S that used multi-layer convolution neural networks (CNN) for encoding and decoding. Each convolution layer was parameterized as a large matrix $W \in \mathbb{R}^{2d \times kd}$ and $b_W \in \mathbb{R}^{2d}$, where k was the kernel width and d was the embedding size. The input to convolution layer would be $X \in \mathbb{R}^{kd}$, which was the concatenation of either the input to encoder(decoder) or the output of previous convolution layer. The original paper also adopted gated linear units as non-linearity: $v([A; B]) = A \otimes \sigma(B)$ where $[A; B]$ was the output of convolution and $\otimes$ was point-wise multiplication.

Besides convolutional encoder and decoder, Gehring et al. (2017) used a special attention mechanism called multi-step attention. It was essentially taking global attention for every decoder layer. At each decoding step, for layer l, the decoder first combined previous output word and convolution output similar to an RNN decoder: $d_t^l = W_d^l h_t^l + b_d^l + g_t$ where h_t^l was the output of convolutional layer and g_t was the previous target word. The model then computed dot product global attention of d_t^l with respect to output of last layer of encoder. The context was computed as weighted sum of the sum of input sentences and encoder's final layer output: $c_t^l = \sum_j \alpha_{tj}^l (z_j + e_j)$. Finally, c_t^l was added back to decoder's states as final decoder output of layer l.

This paper adopted the basic principle of RNN encoder-decoder model with attention, while facilitated parallel computation on GPU. To further equip the model with a sense of time step, Gehring et al. (2017) added positional embedding to original input word embedding. In additional, residual connections were added to enable deep networks.

Transformer

Vaswani et al. (2017) proposed a model that adopted very similar design philosophy of ConvS2S (Gehring et al., 2017), called the transformer. The goal of transformer was also to reduce sequential computation while reserved the elegance of encoder-decoder structure and effectiveness of attention mechanism. In fact, by using only attention mechanism for encoding and decoding, the transformer achieved not only great training speed but also state-of-the-art translation performance.

Vaswani et al. (2017) took advantage of a general view of attention: an attention function is a mapping of a query and a set of key-value pairs to an output value, where the output is computed as a weighted sum of values and weights are computed by query and keys. In the original paper, the particular attention was called scaled dot-product attention, computed as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (6)$$

where matrix Q contains packed queries, matrices K and V are key-value pairs and d_k is a magic scaling factor. To enable the model to attend to different semantics, Vaswani et al. (2017) further proposed multi-head attention, which linearly projected each query, key, value into smaller vector space for attention respectively. The results were then concatenated back.

Similar to Gehring et al. (2017), the encoder of transformer was composed of multiple layers. Each layer contained a self-attention sub-layer and a position-wise feed forward network sub-layer. Self-attention was the attention function with queries, keys and values being all the same i.e. $Attention(X, X, X)$, which functioned similarly to CNN of ConvS2S. The output of self-attention then went to a feed forward network for non-linearity.

The decoder of transformer shared the same sub-layers as the encoder. However, the decoder had a unique sub-layer, which performed multi-head attention over the output of the encoder stack. This sub-layer essentially took global attention of each decoder state with respect to encoder's output and functioned very similar to multi-step attention of Gehring et al. (2017).

In addition to the aforementioned similarities, the transformer also adopted position embedding and residual connections for the same reason as Gehring et al. (2017). The advantages of the transformer are two-folds. First, attention functions can be greatly parallelized. Second, self-attention allows the model to establish dependencies to arbitrarily distant words in input sentence in constant amount of operations.

3 Supervised Learning for NMT Models

The training settings of supervised learning for NMT models is not very interesting. Most NMT models trained with the same objective, which is

to maximize the conditional probability of $P(\mathbf{y} \mid \mathbf{x})$. Since decoders of NMT models generate target word distribution one at a time, the conditional probability is further decomposed into $P(\mathbf{y} \mid \mathbf{x}) = \prod_t P(y_t \mid y_{t-1}, \ldots, y_1, \mathbf{x})$. Hence, the general training goal for NMT model, particularly attentional model, is to maximize the log likelihood on N parallel sentence pairs,

$$\jmath(\theta) = \sum_{n=1}^{N} \sum_{t} \log P(y_t^{(n)} \mid \mathbf{y}_{<t}^{(n)}, h_{t-1}^{(n)}, f^{att}; \theta) \tag{7}$$

where $h_{t-1}^{(n)}$ denotes an internal decoder state and f^{att} denotes the attention mechanism for the model (Hassan et al., 2018).

4 Semi-Supervised Learning for NMT Models

Semi-supervised learning for NMT models is very useful since large parallel corpora is hard to obtain while monolingual corpora is easy to collect. Moreover, with well designed learning methods, monolingual data can be used without modifying original models, which could help boost the performance of well trained NMT models. The most important idea for semi-supervised learning in NMT is the duality of translation task and back translation (Sennrich et al., 2016; He et al., 2016; Hassan et al., 2018).

Sennrich et al. (2016) first proposed two general learning framework for semi-supervised learning for NMT models that required no modification to the original model structure. The first method was to provide target monolingual training examples with empty source sentence, which was essentially training a language model. The second method alleviated the drawback by providing target monolingual training examples with synthetic source (back translation). The source was obtained through a separately trained translation model. In the original paper's experiment, although back translation helped boost the model's performance, there was concern that the quality of back translation might be a bottleneck.

It is easy to observe that back translation is facilitated by a more general property of translation task, duality. For any translation task i.e. En to Zh, there exists a dual task i.e. Zh to En. He et al. (2016) went on utilizing monolingual data of both source side and target side by dual learning. Dual learning formulated the learning problem as a two-agent communication game, where agents were well trained language models i.e. LM_A and LM_B and communication channels were two translation models i.e. P_{AB} and P_{BA}.

The game of dual learning began with one agent translated a source sentence s through translation model P_{AB} into s_{mid} to the other agent. The other agent would give a translation reward by $LM_B(s_{mid})$ and back translated s_{mid} with a back translation reward $\log P_{BA}(s_{mid})$. The models were then updated with reinforcement learning (He et al., 2016).

The dual property of translation task offers more possibilities for semi-supervised learning. Recently, Zhang et al. (2018) took advantage of dual property but formulated the learning process in a different way from dual learning (He et al., 2016). Instead of training two language models, Zhang et al. (2018) updated models with a joint EM optimization method.

Given parallel corpus $D = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ and monolingual corpus $Y = \{(\mathbf{y}^{t})\}_{t=1}^{T}$, the objective for semi-supervised learning was

$$L^*(\theta_{xy}) = \sum_{n} \log P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}) + \sum_{t} \log P(\mathbf{y}^{(t)}) \tag{8}$$

where the second half of the right hand side could be maximized by EM algorithm,

$$\log P(\mathbf{y}^{(t)}) = \log \sum_{x} Q(x) \frac{P(x, \mathbf{y}^{(t)})}{Q(x)} \tag{9}$$

$$\geq \sum_{x} Q(x) \log \frac{P(x, \mathbf{y}^{(t)})}{Q(x)} \tag{10}$$

To make the equal sign valid, we let $Q(\mathbf{x}) = P^*(\mathbf{x} \mid \mathbf{y}^{(t)})$. Since $P^*(\mathbf{x} \mid \mathbf{y}^{(t)})$ was intractable, Zhang et al. (2018) approximated this with back translation $P(\mathbf{x} \mid \mathbf{y}^{(t)})$ and trained the model with new objective as an lower bound of $L^*(\theta_{xy})$:

$$L(\theta_{xy}) = \sum_{n} \log P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}) + \sum_{t} \sum_{x} P(\mathbf{x} \mid \mathbf{y}^{(t)}) \log P(\mathbf{y}^{(t)} \mid \mathbf{x})$$

5 Unsupervised Learning for NMT Models

Recently, several researchers proposed to use cross-lingual word embeddings as a starting point to enable unsupervised neural machine translation

(Artetxe et al., 2017; Lample et al., 2017). A shared encoder would be used to transform sentences into language independent representation. The training process not only resembled the training of denoising autoencoder and but also leveraged back translation to boost the models' ability to perform translation.

While Artetxe et al. (2017) used a dual approach to iteratively update the model, Lample et al. (2017) adopted adversarial training to update the model. Both work showed the possibility of unsupervised learning for NMT models, but experiments of Artetxe et al. (2017) showed that the model structure might still need improvements.

6 Future Developments

This survey discussed the structure of some NMT models from the simplest RNN encoder-decoder to the state-of-the-art transformer. It further investigated different training settings for neural machine translation i.e. supervised, semi-supervised and unsupervised, among which semi-supervised learning could boost NMT models' performance with monolingual data without modifying the models themselves.

Despite the success of encoder-decoder architecture, it's possible to go beyond encoder-decoder. Hassan et al. (2018) used an deliberation network as polish process, which is a close analogy to human translator. And neural machine translation beyond one pass decoding might further improve the performance of current NMT models.

Acknowledgments

This survey is for the project of COMP6122B in Spring 2018.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *CoRR*, abs/1710.11041.

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, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1243–1252.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *CoRR*, abs/1803.05567.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 820–828.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *CoRR*, abs/1711.00043.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *CoRR*, abs/1703.01619.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-*

13 2014, Montreal, Quebec, Canada, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.

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.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Joint training for neural machine translation models with monolingual data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.