Literature Review on

Neural Machine Translation

Nikita Das, M.Sc. Big Data Analytics, Jai Hind College (Autonomous)

Abstract: In the recent decades (1980s – 1990s), researchers have started showing interest in the field of Machine Translation. Initially a statistical approach, phrase based Statistical Machine Translation was followed which was not an efficient model for complex scenarios. Methods kept on evolving and Neural Machine Translation was introduced which used the concepts of Artificial Neural Networks in the translation of one natural language to another. This literature review consists of comparison and view on various research papers based on Neural Machine Translation and the methodologies adopted in them.

Introduction

Statistical Machine translation used count based models which were used for decades but didn't work well later when the computational resources were exceeded by computational complexities and then the idea was dropped for almost 2 decades. Later in the late 2000s the concept sprung back up, this time with a change in methodology of using a single neural network. The neural language translation was integrated by the statistical translation systems. This made machine translation a useful tool for many applications regarding information gisting and increasing productivity of professional translators.

Moving beyond the use in language models, neural network methods focused into other components of traditional statistical machine translation, such as providing additional scores or extending translation, reordering and pre-ordering models and so on. Neural Machine Translation is widely adopted in the industry and has been a hot topic since the past decade. The following table shows the researches going on in the recent years according to the Google Scholar.

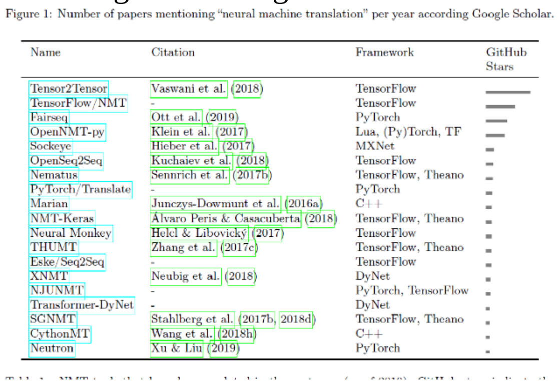


Figure 1: Number of papers mentioning "neural machine translation" per year according Google Scholar.

Name	Citation	Framework	GitHub Stars
Tensor2Tensor	Vaswani et al. (2018)	TensorFlow	▬
TensorFlow/NMT	-	TensorFlow	▬
Fairseq	Ott et al. (2019)	PyTorch	▬
OpenNMT-py	Klein et al. (2017)	Lua, (Py)Torch, TF	▬
Sockeye	Hieber et al. (2017)	MXNet	▬
OpenSeq2Seq	Kuchaiev et al. (2018)	TensorFlow	▬
Nematus	Sennrich et al. (2017b)	TensorFlow, Theano	▬
PyTorch/Translate	-	PyTorch	▬
Marian	Junczys-Dowmunt et al. (2016a)	C++	▬
NMT-Keras	Álvaro Peris & Casacuberta (2018)	TensorFlow, Theano	▬
Neural Monkey	Helcl & Libovický (2017)	TensorFlow	▬
THUMT	Zhang et al. (2017c)	TensorFlow, Theano	▬
Eske/Seq2Seq	-	TensorFlow	▬
XNMT	Neubig et al. (2018)	DyNet	▬
NJUNMT	-	PyTorch, TensorFlow	▬
Transformer-DyNet	-	DyNet	▬
SGNMT	Stahlberg et al. (2017b, 2018d)	TensorFlow, Theano	▬
CythonMT	Wang et al. (2018h)	C++	▬
Neutron	Xu & Liu (2019)	PyTorch	▬

Methodology

The mainstream model framework of neural machine translation is encoder - decoder model. The model uses two different neural networks as encoder and decoder respectively. The encoder is used to read the source sentence, encode it into a vector of fixed dimension, and then the decoder reads the vector and generates the corresponding target language sequence. Neural networks used in the current mainstream encoder and decoder models are cyclic neural networks, their variants and convolutional neural networks. However, when the input sentence is long, the vector with fixed dimensions is difficult to store enough information, so the academic circle introduces an attention mechanism to solve this problem. The attention mechanism allows the decoder to look up words or fragments of the input sentence at any time, so it is not necessary for the intermediate vector to store all information.

Embedding

Each word in the input layer is represented by a real vector, which is called "word embedding." Word embedding can be understood as embedding a vocabulary into a real space of a fixed dimension. Converting a word number into a word vector reduces the input dimension, reduces the parameters and statistics of the cyclic neural network, and the word vector converts the sparse number into a vector of chips, so that the word vector can contain more information.

1. **Transformer Model Architecture**
   It is encoder-decoder structure model where model encodes the input sequence into some encoded sequence. The decoder generates the output sequence from the encoded sequence. In each step, the model is autoregressive [7], consuming the previously generated symbols as additional inputs when generating the next one. Transformer follows this overall architecture, using a stacking layer of self-focused and fully point-by-point encoders and decoders.
   The encoder is made up of the same 6 modules, each of which has two sublayers. The first sublevel is the Multi-Head self-attention mechanism, where self-attention indicates that both the input and output sequences are the same. The second sub-level uses a fully connected network, the main role is to pay attention to the characteristics of the sub-level. In addition, each sub-level adds a residual connection and hierarchical normalization. The decoder is also composed of the same six modules. Each decoder module has three sub-layers. Each sub-layer also adds residual connection and hierarchical normalization. The first and third sublayers are identical to the encoder's Multi-Head Self-Attention Layer and the Full Connection Layer, respectively, and the Multi-Head Attention mechanism used by the second sub-layer uses the encoder's output as Key and Value. The output of the first sublayer of the decoding module is used as the Query.

2. RNN model

Recurrent neural networks (RNN) are a class of neural networks that are helpful in modelling sequence

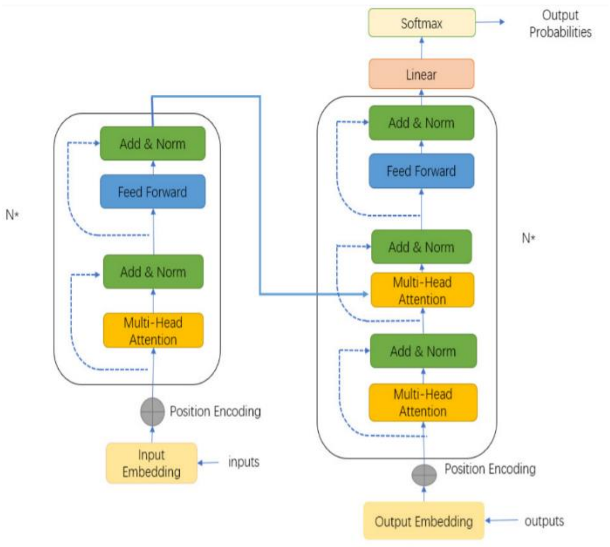


Figure 1: The Transformer model architecture

data. Derived from feedforward networks, RNNs exhibit similar behaviour to how human brains function. Simply put: recurrent neural networks produce predictive results in sequential data that other algorithms can't.
Bi-RNN can be used. BiRNN runs one RNN in the forward direction and another RNN in the other direction, and connects the two results. One of the benefits of using an RNN network is the ability to learn the sequence of sequences so that positional embedding can be used instead.

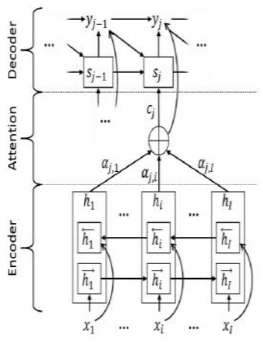
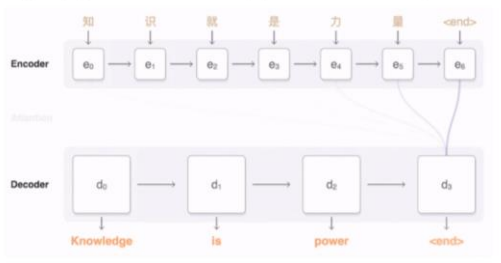


Figure 10: Illustration of the attention mechanism in RNNsearch (Bahdanau

The concept of attention to avoid having a fixed-length source sentence representation. Their model does not use a constant context vector c(x) any more which encodes the whole source sentence. By contrast, the attentional decoder can place its attention only on parts of the source sentence which are useful for producing the next token. The constant context vector c(x) is thus replaced by a series of context vectors cj(x); one for each time step j.

3. **Google Neural Machine Translation system (GNMT)**
   First, the network encodes the source words as a list of vectors, where each vector represents the meaning of all words read so far ("Encoder"). Once the entire sentence is read, the decoder begins, generating the desired sentence one word at a time ("Decoder"). To generate the translated word at each step, the decoder pays attention to a weighted distribution over the encoded source vectors most relevant to generate the desired word ("Attention" represents how much the decoder pays attention to an encoded word).
   Using human-rated side-by-side comparison as a metric, the GNMT system produces translations that are vastly improved compared to the previous phrase-based production system. GNMT reduces translation errors by more than 55%-85% on several major language pairs measured on sampled sentences from Wikipedia and news websites with the help of bilingual human raters.



4. **RNN to Transformers**
   The difference between Transformer and RNN includes multiple source attention layers, multi-head attention, layer normalization and the residual upscaling feed-forward layers. The modules in our Transformer model introduce a cyclic neural network to improve the circulating neural network.
   The encoder consists of a layer of bi-RNN and a stack of six identical layers. Of the six identical layers, each layer is divided into two sub-layers, the first sub-layer is the RNN network, and the second sub-layer is a simple, fully connected feedforward network. The paper use residual connections around both sublayers and then normalize the layers.
   The decoder consists of six identical layer stacks, one layer of attention and one layer of feedforward network. In addition to the two sublayers in each encoder layer, the decoder uses two additional sublayers that perform multi-head attention on the output of the encoder stack. Similar to the encoder, The model use a residual connection around each sublayer and then normalize the layer.

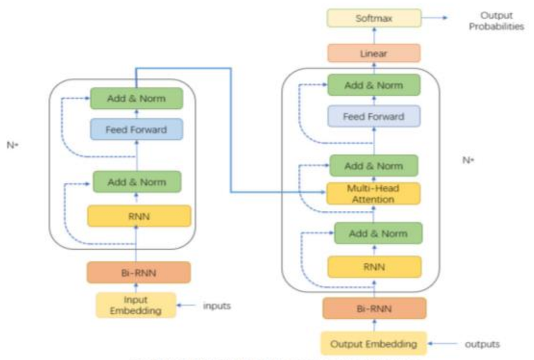


Fig.2 The RNN To Transformer model architecture

Comparison
Neural machine translation can be done in 3 ways using recurrent neural networks, convolution neural networks and self attention based. In this paper we have discussed the RNN type of NMT. All these models are encoder-decoder based models. They all use attention for connecting the decoder to the encoder. The papers also talk about the fusion of Recurrent NMT and model which uses ideas of transformer like multi-head attention to RNMT. The methodology discussed in this paper is the most recent experiment by Hao et al. (2019) who used a Transformer encoder and a recurrent encoder in parallel. This model is far more effective than just the RNN model.

Conclusion

We reviewed the most commonly used building blocks of NMT architectures – recurrence, convolution, and attention – and discussed popular concrete architectures such as RNNsearch, GNMT and the Transformer. The topic is still under study as we can see that the recent experiments are been conducted in the year 2019 and there are many more turns to come.

References

[1] https://arxiv.org
[2] https://www.researchgate.net/publication
[3] https://jair.org/index.php/jair/article/