VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

INFORMATION TECHNOLOGY FACULTY

INTRODUCTION TO
DEEP LEARNING

HOMEWORK

Attention Mechanism

Instructor: **Prof. Lê Anh Cường**

Name: **Đỗ Phạm Quang Hưng**

Student ID: **520K0127**

HO CHI MINH CITY, 2023

Table of Contents

Attention in Long Short-Term Memory Recurrent Neural Networks

The Encoder-Decoder architecture is popular because it has demonstrated state-of-the-art results across a range of domains.

A limitation of the architecture is that it encodes the input sequence to a fixed length internal representation. This imposes limits on the length of input sequences that can be reasonably learned and results in worse performance for very long input sequences.

In this post, you will discover the attention mechanism for recurrent neural networks that seeks to overcome this limitation.

Problem With Long Sequences

The encoder-decoder recurrent neural network is an architecture where one set of LSTMs learn to encode input sequences into a fixed-length internal representation, and a second set of LSTMs read the internal representation and decode it into an output sequence.

This architecture has shown state-of-the-art results on difficult sequence prediction problems like text translation and quickly became the dominant approach.

For example, see:

- Sequence to Sequence Learning with Neural Networks, 2014
- Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, 2014

The encoder-decoder architecture still achieves excellent results on a wide range of problems. Nevertheless, it suffers from the constraint that all input sequences are forced to be encoded to a fixed length internal vector.

This is believed to limit the performance of these networks, especially when considering long input sequences, such as very long sentences in text translation problems.

> *"A potential issue with this encoder–decoder approach is that a neural network needs to be able to compress all the necessary information of a source sentence into a fixed-length vector. This may make it difficult for the neural network to cope with long sentences, especially those that are longer than the sentences in the training corpus."*

— Dzmitry Bahdanau, et al., Neural machine translation by jointly learning to align and translate, 2015

Attention within Sequences

Attention is the idea of freeing the encoder-decoder architecture from the fixed-length internal representation.

This is achieved by keeping the intermediate outputs from the encoder LSTM from each step of the input sequence and training the model to learn to pay selective attention to these inputs and relate them to items in the output sequence.

Put another way, each item in the output sequence is conditional on selective items in the input sequence.

> *“Each time the proposed model generates a word in a translation, it (soft-)searches for a set of positions in a source sentence where the most relevant information is concentrated. The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words.*
>
> *… it encodes the input sentence into a sequence of vectors and chooses a subset of these vectors adaptively while decoding the translation. This frees a neural translation model from having to squash all the information of a source sentence, regardless of its length, into a fixed-length vector.”*

— Dzmitry Bahdanau, et al., Neural machine translation by jointly learning to align and translate, 2015

This increases the computational burden of the model, but results in a more targeted and better-performing model.

In addition, the model is also able to show how attention is paid to the input sequence when predicting the output sequence. This can help in understanding and diagnosing exactly what the model is considering and to what degree for specific input-output pairs.

> *“The proposed approach provides an intuitive way to inspect the (soft-)alignment between the words in a generated translation and those in a source sentence. This is done by visualizing the annotation weights… Each row of a matrix in each plot*

> *indicates the weights associated with the annotations. From this we see which positions in the source sentence were considered more important when generating the target word."*

— Dzmitry Bahdanau, et al., Neural machine translation by jointly learning to align and translate, 2015

How Does Attention Work in Encoder-Decoder Recurrent Neural Networks

Encoder-Decoder Model

The Encoder-Decoder model for recurrent neural networks was introduced in two papers.

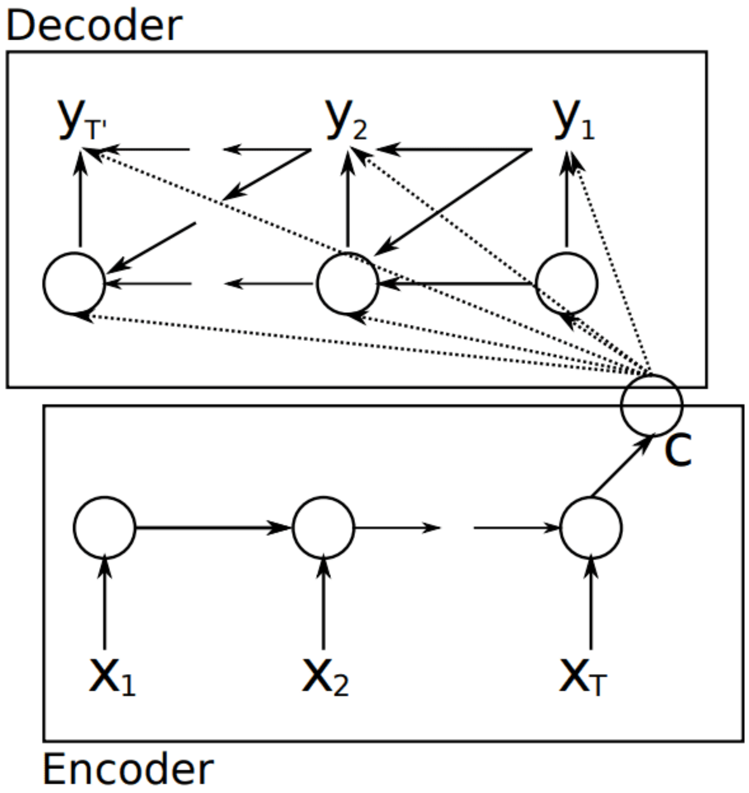
Both developed the technique to address the sequence-to-sequence nature of machine translation where input sequences differ in length from output sequences.

Ilya Sutskever, et al. do so in the paper "Sequence to Sequence Learning with Neural Networks" using LSTMs.

Kyunghyun Cho, et al. do so in the paper "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". This work, and some of the same authors (Bahdanau, Cho and Bengio) developed their specific model later to develop an attention model. Therefore we will take a quick look at the Encoder-Decoder model as described in this paper.

From a high-level, the model is comprised of two sub-models: an encoder and a decoder.

- Encoder: The encoder is responsible for stepping through the input time steps and encoding the entire sequence into a fixed length vector called a context vector.
- Decoder: The decoder is responsible for stepping through the output time steps while reading from the context vector.



Encoder-Decoder Recurrent Neural Network Model.

Taken from "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation"

> *"We propose a novel neural network architecture that learns to encode a variable-length sequence into a fixed-length vector representation and to decode a given fixed-length vector representation back into a variable-length sequence*."

— Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, 2014.

Key to the model is that the entire model, including encoder and decoder, is trained end-to-end, as opposed to training the elements separately.

The model is described generically such that different specific RNN models could be used as the encoder and decoder.

Instead of using the popular Long Short-Term Memory (LSTM) RNN, the authors develop and use their own simple type of RNN, later called the Gated Recurrent Unit, or GRU.

Further, unlike the Sutskever, et al. model, the output of the decoder from the previous time step is fed as an input to decoding the next output time step. You can see this in the image above where the output y2 uses the context vector (C), the hidden state passed from decoding y1 as well as the output y1.

> *"… both y(t) and h(i) are also conditioned on y(t−1) and on the summary c of the input sequence."*

— Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, 2014

Attention Model

Attention was presented by Dzmitry Bahdanau, et al. in their paper "Neural Machine Translation by Jointly Learning to Align and Translate" that reads as a natural extension of their previous work on the Encoder-Decoder model.

Attention is proposed as a solution to the limitation of the Encoder-Decoder model encoding the input sequence to one fixed length vector from which to decode each output time step. This issue is believed to be more of a problem when decoding long sequences.

> *"A potential issue with this encoder–decoder approach is that a neural network needs to be able to compress all the necessary information of a source sentence into a fixed-length vector. This may make it difficult for the neural network to cope with long sentences, especially those that are longer than the sentences in the training corpus."*

— Neural Machine Translation by Jointly Learning to Align and Translate, 2015.

Attention is proposed as a method to both align and translate.

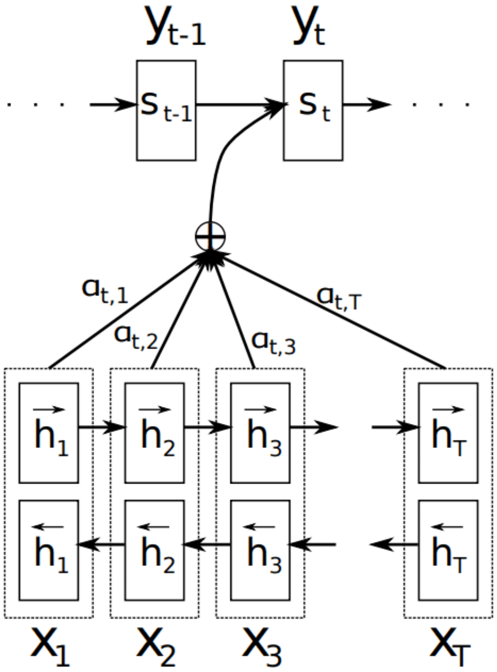
Alignment is the problem in machine translation that identifies which parts of the input sequence are relevant to each word in the output, whereas translation is the process of using the relevant information to select the appropriate output.

> *"… we introduce an extension to the encoder–decoder model which learns to align and translate jointly. Each time the proposed model generates a word in a translation, it (soft-)searches for a set of positions in a source sentence where the most relevant information is concentrated. The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words."*

— Neural Machine Translation by Jointly Learning to Align and Translate, 2015.

Instead of encoding the input sequence into a single fixed context vector, the attention model develops a context vector that is filtered specifically for each output time step.

As with the Encoder-Decoder paper, the technique is applied to a machine translation problem and uses GRU units rather than LSTM memory cells. In this case, a bidirectional input is used where the input sequences are provided both forward and backward, which are then concatenated before being passed on to the decoder.



Example of Attention

Taken from "Neural Machine Translation by Jointly Learning to Align and Translate", 2015.

Extensions to Attention

This section looks at some additional applications of the Bahdanau, et al. attention mechanism.

Hard and Soft Attention

In the 2015 paper "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", Kelvin Xu, et al. applied attention to image data using convolutional neural nets as feature extractors for image data on the problem of captioning photos.

They develop two attention mechanisms, one they call "soft attention," which resembles attention as described above with a weighted context vector, and the second "hard attention" where the crisp decisions are made about elements in the context vector for each word.

They also propose double attention where attention is focused on specific parts of the image.

Dropping the Previous Hidden State

There have been some applications of the mechanism where the approach was simplified so that the hidden state from the last output time step (s(t-1)) is dropped from the scoring of annotations (Step 3. above).

Two examples are

- Hierarchical Attention Networks for Document Classification, 2016.
- Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification, 2016

This has the effect of not providing the model with an idea of the previously decoded output, which is intended to aid in alignment.

This is noted in the equations listed in the papers, and it is not clear if the mission was an intentional change to the model or merely an omission from the equations. No discussion of dropping the term was seen in either paper.

Study the Previous Hidden State

Minh-Thang Luong, et al. in their 2015 paper “Effective Approaches to Attention-based Neural Machine Translation” explicitly restructure the use of the previous decoder hidden state in the scoring of annotations. Also, see the presentation of the paper and associated Matlab code.

They developed a framework to contrast the different ways to score annotations. Their framework calls out and explicitly excludes the previous hidden state in the scoring of annotations.

Instead, they take the previous attentional context vector and pass it as an input to the decoder. The intention is to allow the decoder to be aware of past alignment decisions.

> *“… we propose an input-feeding approach in which attentional vectors ht are concatenated with inputs at the next time steps […]. The effects of having such connections are two-fold: (a) we hope to make the model fully aware of previous alignment choices and (b) we create a very deep network spanning both horizontally and vertically.”*

— Effective Approaches to Attention-based Neural Machine Translation, 2015.

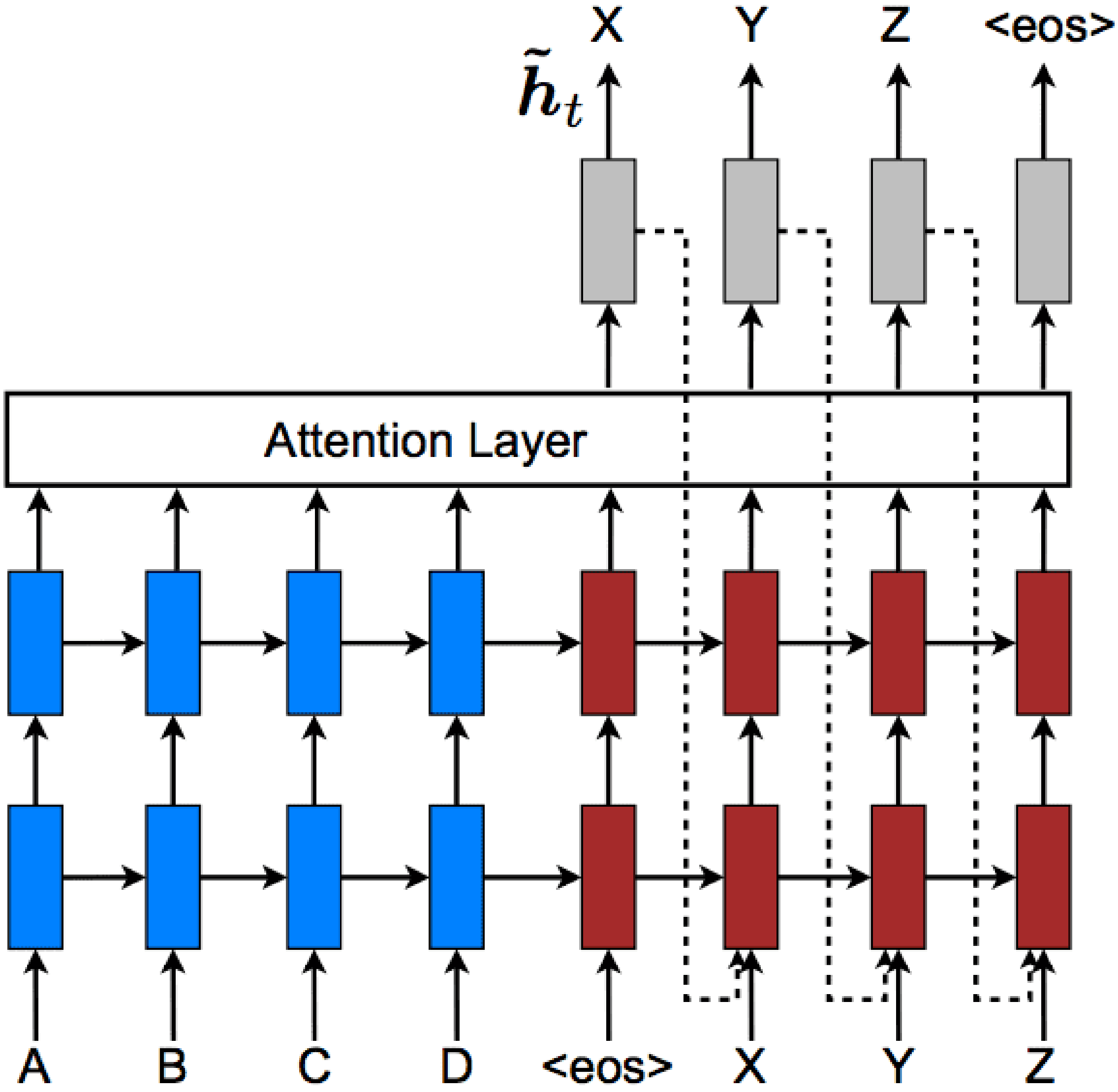
They also develop “global” vs “local” attention, where local attention is a modification of the approach that learns a fixed-sized window to impose over the attentional vector for each output time step. It is seen as a simpler approach to the “hard attention” presented by Xu, et al.

> *“The global attention has a drawback that it has to attend to all words on the source side for each target word, which is expensive and can potentially render it impractical to translate longer sequences, e.g., paragraphs or documents. To address this deficiency, we propose a local attentional mechanism that chooses to focus only on a small subset of the source positions per target word.”*

— Effective Approaches to Attention-based Neural Machine Translation, 2015.

Analysis in the paper of global and local attention with different annotation scoring functions suggests that local attention provides better results on the translation task.

Below is a picture of this approach taken from the paper. Note the dotted lines explicitly showing the use of the decoders attended hidden state output (ht) providing input to the decoder on the next timestep.

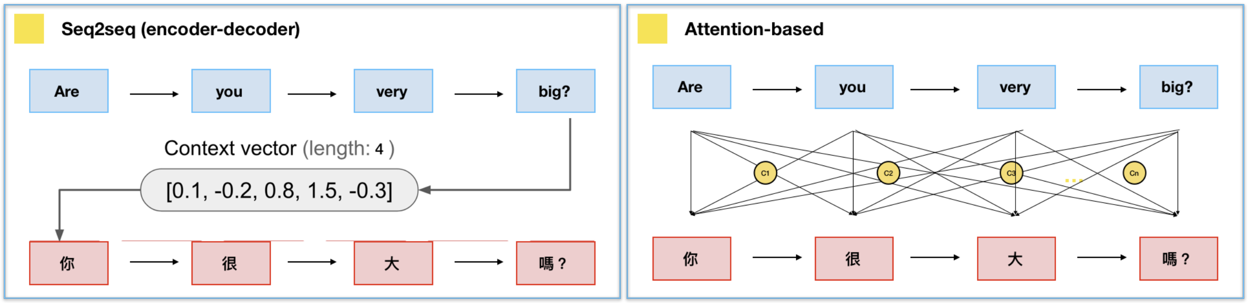


Feeding Hidden State as Input to Decoder

Taken from "Effective Approaches to Attention-based Neural Machine Translation", 2015.

Attention Mechanism For Machine Translation Task

In machine translation, the encoder-decoder architecture is common. The encoder reads a sequence of words and represents it with a high-dimensional real-valued vector. This vector, often called the context vector, is given to the decoder, which then generates another sequence of words in the target language. If the input sequence is very long, a single vector from the encoder doesn't give enough information for the decoder.



Context vectors (right) carry attention information from encoder to decoder.

Source: Su 2018, fig. 15.

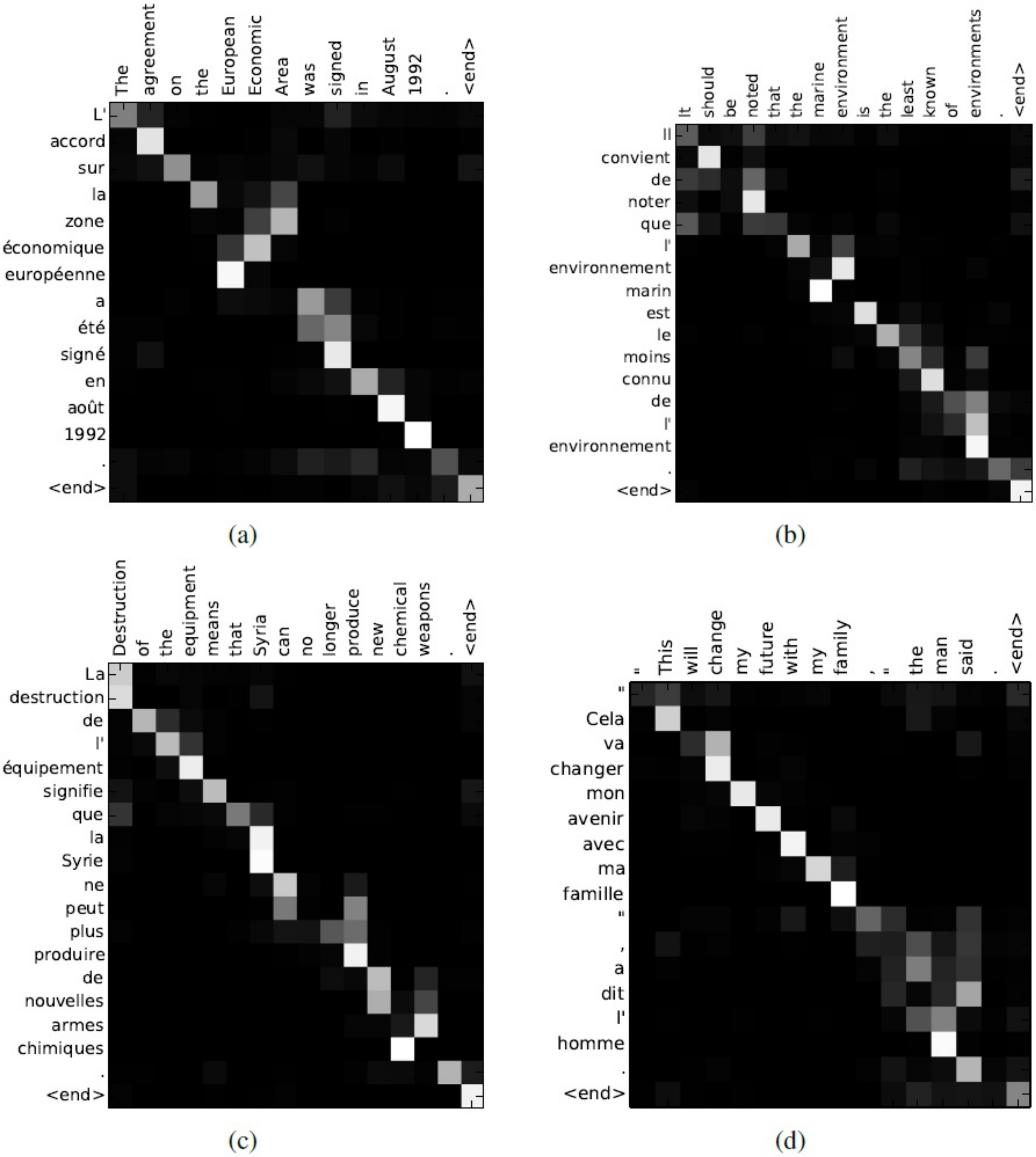
Attention is about giving more contextual information to the decoder. At every decoding step, the decoder is informed how much "attention" it should give to each input word. While attention started this way in sequence-to-sequence modeling, it was later applied to words within the same sequence, giving rise to self-attention and transformer architecture.

Since the late 2010s, the attention mechanism has become popular, sometimes replacing CNNs, RNNs and LSTMs.

Attention In a Picture

Consider an example from machine translation. The sentence "The agreement on the European Economic Area was signed in August 1992" is to be translated to French, which might be "L'accord sur la zone économique européenne a été signé en août 1992". We can see that "Economic" becomes "économique" and "European" becomes "européenne", but their positions are swapped. The phrase "was signed" becomes "a été signé". Thus, translation depends not just on individual words but also their context within the sentence. Attention is meant to capture this context.

In this example, attention is passed from the encoder to the decoder. The decoder generates the translated words one by one. Each output word is influenced by all input words in different amounts. Attention captures these weights.

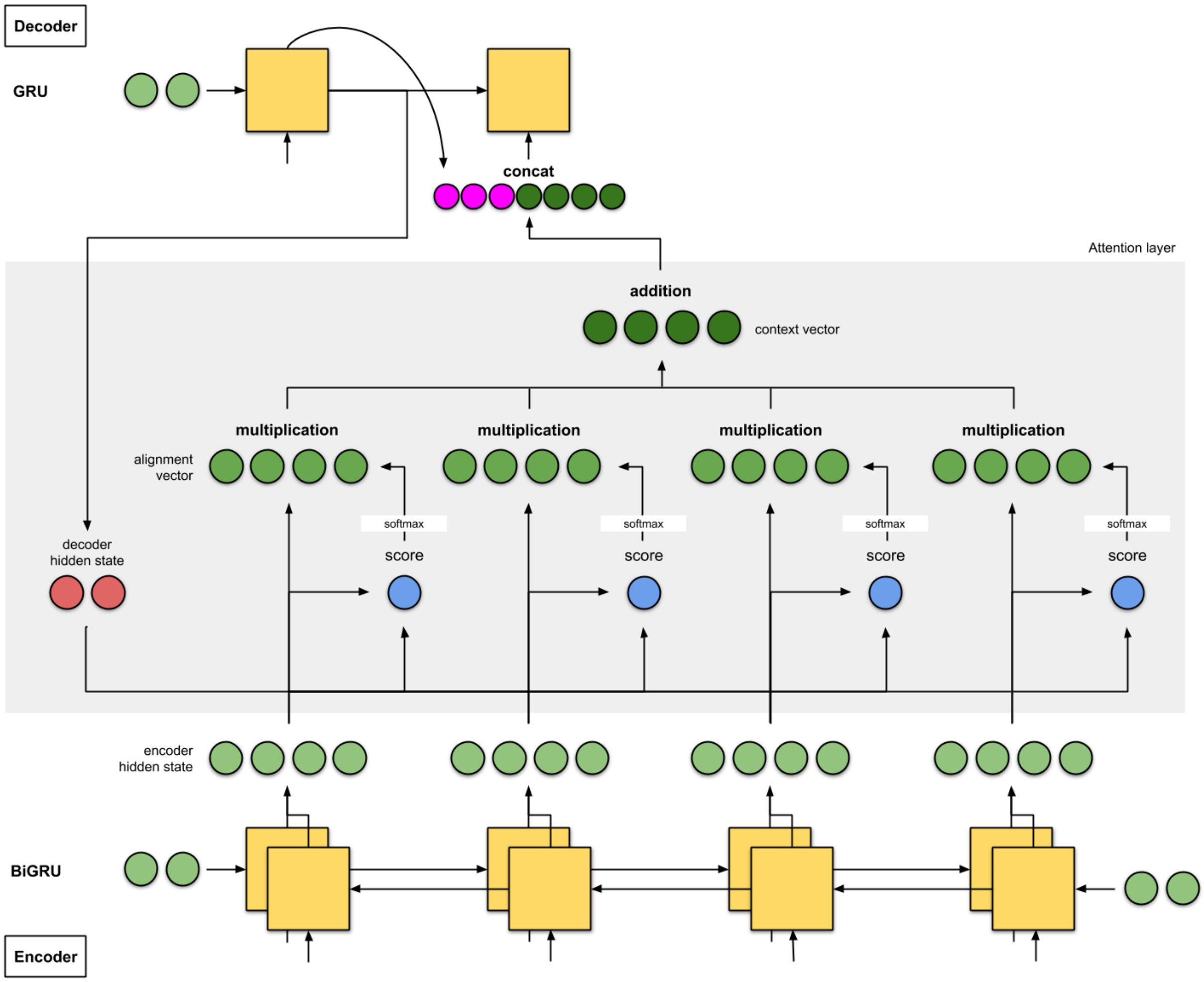


Heatmap showing attention between source and target languages. Source: Bahdanau et al. 2016, fig. 3.

We can also visualize attention via heatmaps. In the figure, we map English words to translated French words. We note that sometimes a translated word is attended to by multiple English words. Lighter colors represent higher attention.

Architecture of Attention

Let's consider machine translation as explained by Bahdanau et al. (2014). Encoder is a bidirectional RNN while the decoder is an RNN. The input sequence is fed into the encoder whose hidden states are exposed to the decoder via the attention layer. More specifically, the backward and forward hidden encoder states are concatenated. These states are weighted to give a context vector that's used by the decoder. Attention weights are calculated by aligning the decoder's last hidden state with the encoder's hidden states.



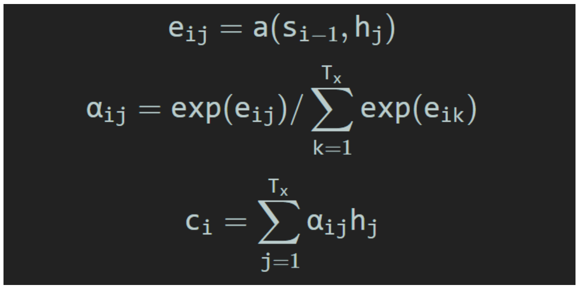
Attention is calculated from hidden states of encoder and recent hidden state of decoder. Source: Karim 2019.

The decoder's current hidden state is a function of its previous hidden state, previous output word and the context vector. Attention is passed via the context vector, which itself is based on the alignment of encoder and decoder states.

Luong et al. proposed a slightly different architecture. Their encoder and decoder are each a 2-layer LSTM. It also uses a feedforward network for the final output. In Google's Neural Machine Translation, 8-layer LSTM is used in encoder and decoder. The first encoder layer is bidirectional. Both encoder and decoder include some residual connections.

Alignment Score for Attention

Bahdanau et al. align the decoder's sequence with the encoder's sequence. An alignment score quantifies how well output at position i is aligned to the input at position j. The context vector that goes to the decoder is based on the weighted sum of the encoder's RNN hidden states h_j. These weights come from the alignment. Mathematically, given an alignment model a, alignment energy e, context vector c, and weights α, we have:



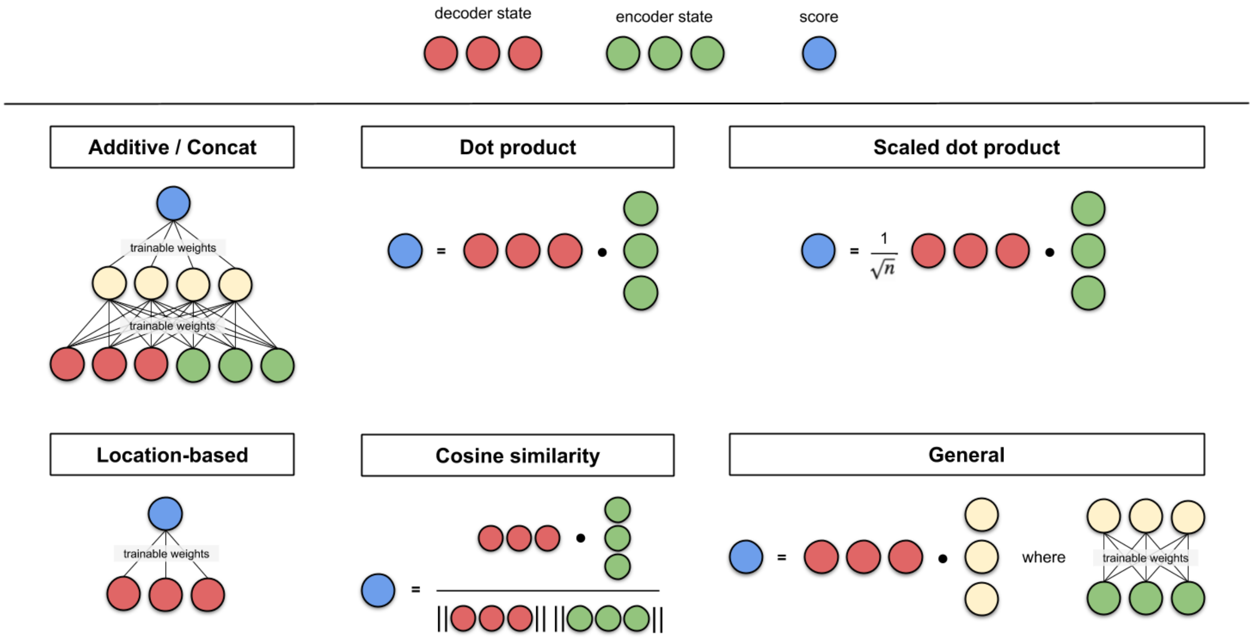
$$e_{ij} = a(s_{i-1}, h_j)$$

$$\alpha_{ij} = \exp(e_{ij}) / \sum_{k=1}^{T_x} \exp(e_{ik})$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

The decoder's hidden state is based on its previous hidden state s_{i-1}, the previous predicted word and the current context vector. At each time step, the context vector is adjusted via the alignment model and attention. Thus, at step step, the decoder selectively attends to the input sequence via the encoder hidden states.

Bahdanau et al. concatenated the forward and backward encoder hidden states and added these with decoder hidden states. Luong et al. proposed many other alternative alignment scores. Vaswani et al. proposed the scaled dot product.



Illustrating different alignment score functions. Source: Karim 2019.

References

1. The Attention Mechanism from Scratch - MachineLearningMastery.com
2. Attention in Long Short-Term Memory Recurrent Neural Networks - MachineLearningMastery.com
3. How Does Attention Work in Encoder-Decoder Recurrent Neural Networks - MachineLearningMastery.com
4. Attention Mechanism in Neural Networks | Developia