African Master's in Machine Intelligence Ghana

The Transformer

From RNN to Attention

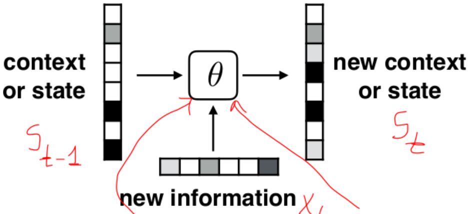
Salomon Kabongo
skabongokabenamualu@acm.org

April 17, 2020

Content

Introduction 1
Seq2Seq Model

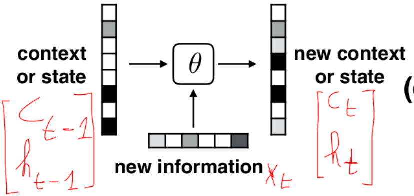
Recurrent Neural Networks (RNN), **Long Short-Term Memory (LSTM)** and **Gated Recurrent neural networks (GRU)** in particular, have been firmly established as state of the art approaches in sequence modeling [4].



$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

$m \times 1$ $\quad$ $m \times m$ $\quad$ $m \times 1$ $\quad$ $m \times d$ $\quad$ $d \times 1$

Figure: RNN



LSTM (one of many)

$$f_t = \text{sigmoid}(W^{f,h} h_{t-1} + W^{f,x} x_t)$$ forget gate

$$i_t = \text{sigmoid}(W^{i,h} h_{t-1} + W^{i,x} x_t)$$ input gate

$$o_t = \text{sigmoid}(W^{o,h} h_{t-1} + W^{o,x} x_t)$$ output gate

what portion of past signal to remember

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W^{c,h} h_{t-1} + W^{c,x} x_t)$$ memory cell

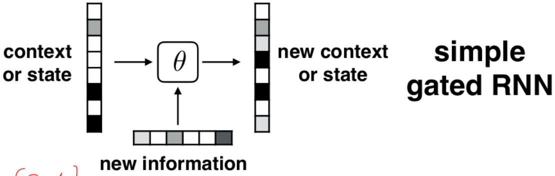
what quantity of input signal to retain

$$h_t = o_t \odot \tanh(c_t)$$ visible state

what portion of tanh(c_t) to keep

Since the memory cell is additive it may increase in value that why wich put it [-1, 1]

Figure: LSTM



$\in [0,1]$

$$g_t = \boxed{\text{sigmoid}}\left(\underline{W^{g,s} s_{t-1} + W^{g,x} x_t}\right)$$

$$s_t = (1 - g_t) \odot s_{t-1} + g_t \odot \underline{\tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)}$$

Have same dimension

$\begin{bmatrix}0\\1\\0\\\vdots\end{bmatrix}$ $\begin{bmatrix}1\\0\\1\\\vdots\end{bmatrix}$

element wise multiplication

Figure: GRU

The seq2seq model was born in the field of language modeling [3]. the idea here, is to transform an **input sequence (source)** to a **new one (target)** and both sequences can be of arbitrary lengths.

The seq2seq model was born in the field of language modeling [3]. the idea here, is to transform an **input sequence (source)** to a **new one (target)** and both sequences can be of arbitrary lengths.
The seq2seq has an **encoder-decoder architecture**, composed of:

- **Encoder** : An encoder processes the input **sequence and compresses the information into a context vector** (also known as sentence embedding) of a fixed length.

The seq2seq model was born in the field of language modeling [3]. the idea here, is to transform an **input sequence (source)** to a **new one (target)** and both sequences can be of arbitrary lengths.
The seq2seq has an **encoder-decoder architecture**, composed of:

- **Encoder** : An encoder processes the input **sequence and compresses the information into a context vector** (also known as sentence embedding) of a fixed length.
- **Decoder** : A decoder is initialized with the context vector to emit the transformed output.

Limitation
Seq2Seq Model

- This representation obtained by the encoder (fixed length vector) is **expected to be a good summary of the meaning of the whole source sequence, but this is not always the case.**

Limitation
Seq2Seq Model

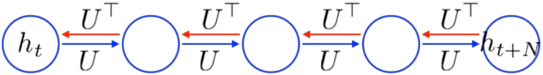
- This representation obtained by the encoder (fixed length vector) is **expected to be a good summary of the meaning of the whole source sequence, but this is not always the case.**
- A critical and apparent disadvantage of this **fixed-length context vector design is incapability of remembering long sentences**.

Looking at the simple RNN naïve transition function

Looking at the simple RNN naïve transition function

$$f\left(h_{t-1}, x_t\right) = \tanh\left(Wx_t + Uh_{t-1} + b\right)$$

With this naïve transition the error must backpropagate through all the intermediate nodes:

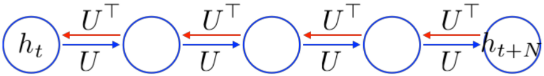


Gated recurrent units to attention
RNN

Looking at the simple RNN naïve transition function

$$f\left(h_{t-1}, x_t\right) = \tanh\left(Wx_t + Uh_{t-1} + b\right)$$

With this naïve transition the error must backpropagate through all the intermediate nodes:



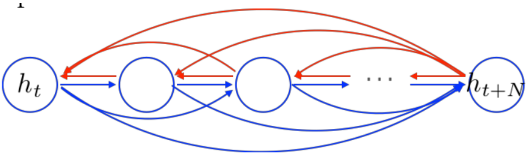
The Back propagation through time imply :

$$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g} \frac{\partial g}{\partial h_{t+N}} \underbrace{\prod_{n=1}^{N} U^\top \operatorname{diag}\left(\frac{\partial \tanh\left(a_{t+n}\right)}{\partial a_{t+n}}\right)}_{\textit{Problematic}!}$$

A key idea behind LSTM and GRU is the additive update.

$$h_t = u_t \odot h_{t-1} + (1 - u_t) \odot \tilde{h}_t, \text{ where } \tilde{h}_t = f(x_t, h_{t-1})$$

This additive update creates linear short-cut connections



What are those adaptive shortcuts [1]?
When unrolled, it's a
weighted combination of all previous hidden vectors.

$$
\begin{aligned}
h_t &= u_t \odot h_{t-1} + (1 - u_t) \odot \tilde{h}_t \\
&= u_t \odot \left(u_{t-1} \odot h_{t-2} + (1 - u_{t-1}) \odot \tilde{h}_{t-1} \right) + (1 - u_t) \odot \tilde{h}_t \\
&= \cdots \\
&= \sum_{i=1}^{t} \left(\prod_{j=i}^{t-i+1} u_j \right) \left(\prod_{k=1}^{i-1} (1 - u_k) \right) \tilde{h}_i
\end{aligned}
$$

1. Can we "free" these dependent weights?
2. Can we "free" candidate vectors?
3. Can we separate keys and values?
4. Can we have multiple attention heads?

$$h_t = \sum_{i=1}^{t} \left(\prod_{j=i}^{t-i+1} u_j \right) \left(\prod_{k=1}^{i-1} (1 - u_k) \right) \tilde{h}_i \quad \mathbf{0}$$

$$h_t = \sum_{i=1}^{t} \alpha_i \tilde{h}_i, \text{ where } \alpha_i \propto \exp(\text{ATT}(\tilde{h}_i, x_t)) \quad \mathbf{1}$$

$$h_t = \sum_{i=1}^{t} \alpha_i f(x_i), \text{ where } \alpha_i \propto \exp(\text{ATT}(f(x_i), x_t)) \quad \mathbf{2}$$

$$h_t = \sum_{i=1}^{t} \alpha_i V(f(x_i)), \text{ where } \alpha_i \propto \exp(\text{ATT}(K(f(x_i)), Q(x_t))) \quad \mathbf{3}$$

$$h_t = [h_t^1; \cdots ; h_t^K], \text{ where } h_t^k = \sum_{i=1}^{t} \alpha_i^k V^k(f(x_i)), \text{ and } \alpha_i^k \propto \exp(\text{ATT}(K^k(f(x_i)), Q^k(f(x_t)))) \quad \mathbf{4}$$

The current formulation of the attention is position-invariant:

$$ATT(A, B, C) == ATT(B, C, A)$$

The current formulation of the attention is position-invariant:

$$ATT(A, B, C) == ATT(B, C, A)$$

The idea is to include some sens of position to the formulation, to account for position and distances between inputs.

The current formulation of the attention is position-invariant:

$$ATT(A, B, C) == ATT(B, C, A)$$

The idea is to include some sens of position to the formulation, to account for position and distances between inputs.

$$h_t^k = \sum_{i=1}^{T} \alpha_i^k V^k \left(f\left(x_i\right) + \boxed{\text{p(i)}} \right)$$

$$\alpha_i^k \propto \exp\left(\text{ATT}\left(K^k\left(f\left(x_i\right) + \boxed{\text{p(i)}} \right), Q^k\left(f\left(x_t\right) + \boxed{\text{p(i)}} \right) \right) \right)$$

The current formulation of the attention is position-invariant:

$$ATT(A, B, C) == ATT(B, C, A)$$

The idea is to include some sens of position to the formulation, to account for position and distances between inputs.

$$h_t^k = \sum_{i=1}^{T} \alpha_i^k V^k \left(f(x_i) + \boxed{\text{p(i)}} \right)$$

$$\alpha_i^k \propto \exp \left(\text{ATT} \left(K^k \left(f(x_i) + \boxed{\text{p(i)}} \right), Q^k \left(f(x_t) + \boxed{\text{p(i)}} \right) \right) \right)$$

The choice of positional embedding $p(i)$ can be obtained from:

- Learned Positional Embedding *[Sukhbataar et al., 2016]*
- Sinusoidal Positional Embedding *[Vaswani et al., 2017]*

- Transformer [4] is a model architecture relying entirely **on an attention (self-attention) mechanism without using sequence-aligned recurrent architecture** to draw global dependencies between input and output.

- Transformer [4] is a model architecture relying entirely **on an attention (self-attention) mechanism without using sequence-aligned recurrent architecture** to draw global dependencies between input and output.
- The **Transformer** follows **seq2seq architecture** but using stacked self-attention and point-wise, fully connected layers for both the encoder and decode.

- Transformer [4] is a model architecture relying entirely **on an attention (self-attention) mechanism without using sequence-aligned recurrent architecture** to draw global dependencies between input and output.
- The **Transformer** follows **seq2seq architecture** but using stacked self-attention and point-wise, fully connected layers for both the encoder and decode.
- The encoder **maps an input sequence** of symbol representations $(x_1, \cdots, x_n)$ to a **sequence of continuous representations** $z = (z_1, \cdots, z_n)$. Given z, the **decoder then generates an output sequence** $(y_1, \cdots, y_m)$ of **symbols one element at a time**

The Transformer was first proposed in the paper **Attention is All You Need**

The Transformer was first proposed in the paper **Attention is All You Need**

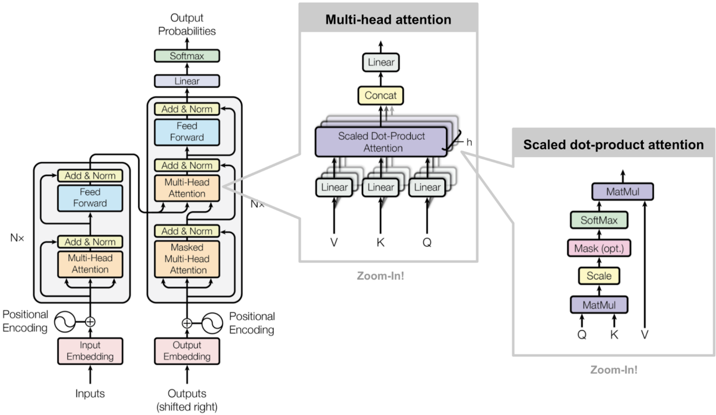


Figure: Image source [5]

The encoder is composed of a stack of $N = 6$ identical layers.

The encoder is composed of a stack of $N = 6$ identical layers.
Each layer has two sub-layers.

- A **multi-head self-attention mechanism**,

The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers.

- A **multi-head self-attention mechanism**,
- A simple, **position-wise fully connected feed-forward network**.

The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers.

- A **multi-head self-attention mechanism**,
- A simple, **position-wise fully connected feed-forward network**.

Each of those sub-layers is as well proceed by a **residual connection** and followed by a **layer norm**.

The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers.

- A **multi-head self-attention mechanism**,
- A simple, **position-wise fully connected feed-forward network**.

Each of those sub-layers is as well proceed by a **residual connection** and followed by a **layer norm**.

The decoder is composed also of a stack of $N = 6$ identical layers.

The decoder is composed also of a stack of $N = 6$ identical layers. Each layer has instead tree sub-layers.

- A **modified multi-head self-attention mechanism**, to prevent positions from attending to subsequent positions.

The decoder is composed also of a stack of $N = 6$ identical layers. Each layer has instead tree sub-layers.

- A **modified multi-head self-attention mechanism**, to prevent positions from attending to subsequent positions.
- A simple, **position-wise fully connected feed-forward network**.

The decoder is composed also of a stack of $N = 6$ identical layers. Each layer has instead tree sub-layers.

- A **modified multi-head self-attention mechanism**, to prevent positions from attending to subsequent positions.
- A simple, **position-wise fully connected feed-forward network**.
- A multi-head attention over the output of the encoder stack

The decoder is composed also of a stack of $N = 6$ identical layers. Each layer has instead tree sub-layers.

- A **modified multi-head self-attention mechanism**, to prevent positions from attending to subsequent positions.
- A simple, **position-wise fully connected feed-forward network**.
- A multi-head attention over the output of the encoder stack

Each of those sub-layers is proceed by a **residual connection** and followed by a **layer norm**.

There have been works to improve the presented **vanilla Transformer** for **longer-term attention span, less memory** [5] and **computation consumption**, ...

References

[1] NLP AMMI Course, Kyunghyun Cho, Facebook AI.

[2] D. Bahdanau, K. Cho, and Y. Bengio.
Neural machine translation by jointly learning to align and translate, 2014.

[3] I. Sutskever, O. Vinyals, and Q. V. Le.
Sequence to sequence learning with neural networks, 2014.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin.
Attention is all you need, 2017.

[5] L. Weng.
The transformer family.
lilianweng.github.io/lil-log, 2020.

TUASAKIDILA