EE-559 – Deep learning

13.1. Attention Mechanisms

François Fleuret

https://fleuret.org/ee559/

Feb 10, 2020
Attention mechanisms aggregate features with an importance score that

- depends on the feature themselves, not only on their position in the tensor,
- relax locality constraints.
Given a 1D convolutional map
\[ x \in \mathbb{R}^{T \times D}, \]
a standard approach to build an aggregated information is average pooling
\[ y_j = \sum_{i=1}^{T} \frac{1_{\{|j-i| \leq \Delta\}}}{\sum_k 1_{\{|j-k| \leq \Delta\}}} x_i. \]

Here the contribution of \( x_i \) to \( y_j \) is entirely driven by their locations [in the tensor]. This is a form of **spatial attention**.
For context attention, the importance of \( x_i \) in computing \( y_j \) is not entirely determined by \( i \) and \( j \) but by \( x_i \) and \( j \) through an attention function \( a(x_i; \theta_j) \)

\[
y_j = \sum_{i=1}^{T} \frac{e^{a(x_i; \theta_j)}}{\sum_{k} e^{a(x_k; \theta_j)}} x_i = \sum_{i=1}^{T} \text{softmax}_i \left( a(x_i; \theta_j) \right) x_i,
\]

where the \( \theta_j \) are model parameters.
We differentiate this from **self-attention**, where the importance of $x_i$ in computing $y_j$ depends on $x_i$ and $x_j$

$$y_j = \sum_{i=1}^{T} \text{softmax}_i (a(x_i, x_j; \theta)) x_i.$$
The most standard approaches implement attention as a dot-product, *e.g.*

\[ a(x; V) = x^T V \]

or

\[ a(x, x'; W, W') = (Wx)^T (W'x') \]

for self-attention.
The most standard approaches implement attention as a dot-product, e.g.

\[ a(x; V) = x^T V \]

or

\[ a(x, x'; W, W') = (Wx)^T (W'x') \]

for self-attention.

However, the quantity \( a \) can take any form, e.g. an MLP.
In what follows there is often an implicit “channel dimension” and the matrix product operates on the channels at every “location” of every sample.

With

\[ W \in \mathbb{R}^{c \times d} \]

and \( x \) a \( d \)-channel tensor

\[ x \in \mathbb{R}^{d \times d \times B} \]

the resulting tensor has same shape with \( c \) channels

\[ Wx \in \mathbb{R}^{d \times c \times B}. \]
Attention for seq2seq
Attention mechanisms were re-introduced for deep learning to provide long-term dependency for sequence-to-sequence translation.
Attention mechanisms were re-introduced for deep learning to provide long-term dependency for sequence-to-sequence translation.

For such a task, given an input sequence $x_1, \ldots, x_T$, the standard approach (Sutskever et al., 2014) is to use a recurrent model

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

and to consider that the final hidden state

$$ v = h_T $$

carries enough information to drive an auto-regressive generative model

$$ y_t \sim p(y_1, \ldots, y_{t-1}, v), $$

itself implemented with another RNN.
The main weakness of such an approach is that all the information has to flow through a single state $v$, whose capacity has to accommodate any situation.

There are no direct “channels” to transport local information from the input sequence to the place where it is useful in the resulting sequence.
Attention mechanisms (Graves et al., 2014; Bahdanau et al., 2014) transport information from parts of the signal to other parts specified dynamically.
Bahdanau et al. (2014) propose to extend a standard recurrent model with such a mechanism. They first run a bi-directionnal RNN to get a hidden state

\[ h_i = (h_i^{\rightarrow}, h_i^{\leftarrow}), \ i = 1, \ldots, T. \]

From this, they compute a new process \( s_i, \ i = 1, \ldots, T \) which looks at weighted averages of the \( h_j \), where the weight are functions of the signal.
Given $y_1, \ldots, y_{i-1}$ and $s_1, \ldots, s_{i-1}$ first compute an attention

$$\forall j, \ e_{i,j} = a(s_{i-1}, h_j)$$

where $a$ is a one hidden layer tanh MLP (this is “additive attention”, or “concatenation”).
Given $y_1, \ldots, y_{i-1}$ and $s_1, \ldots, s_{i-1}$ first compute an attention

$$\forall j, \quad e_{i,j} = a(s_{i-1}, h_j)$$

where $a$ is a one hidden layer tanh MLP (this is “additive attention”, or “concatenation”).

Normalize it into a distribution with a standard softmax

$$\forall j, \quad \alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T} \exp(e_{i,k})},$$

and compute the context vector from the $h$s to predict $y_i$

$$c_i = \sum_{j=1}^{T} \alpha_{i,j} h_j.$$
The model can now make the prediction

\[ s_i = f(s_{i-1}, y_{i-1}, c_i) \]
\[ y_i \sim g(y_{i-1}, s_i, c_i) \]

where \( f \) is a GRU (Cho et al., 2014).
$x_1, \ x_2, \ x_3, \ x_4, \ \ldots, \ x_{T-1}, \ x_T$
RNN
Francois Fleuret
EE-559 – Deep learning / 13.1. Attention Mechanisms
A diagram showing the flow of information in an RNN with attention mechanisms. The input sequence \( x_1, x_2, x_3, \ldots, x_T \) is fed into the RNN, which outputs \( h_1, h_2, h_3, \ldots, h_T \). Attention weights \( \alpha_{3,1} \) are shown connecting the hidden state \( h_3 \) to the input \( x_1 \). The output sequence \( y_1, y_2 \) is influenced by the attention mechanism, with weights \( s_1, s_2 \).
The diagram illustrates the attention mechanism in an RNN. The attention weights $\alpha_{3,1}$, $\alpha_{3,2}$, and $\alpha_{3,T-1}$ are shown as influencing the hidden state transitions $s_1 \rightarrow s_2 \rightarrow s_3$. The input sequence $x_1, x_2, x_3, x_4, \ldots, x_{T-1}, x_T$ is processed through the RNN, and the output sequence $y_1, y_2$ is generated based on the computed attention weights.
Figure 2: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. The results are on the full test set which includes sentences having unknown words to the models.

(Bahdanau et al., 2014)
The agreement on the European Economic Area was signed in August 1992.


It should be noted that the marine environment is the least known of environments.

Il convient de noter que l'environnement marin est le moins connu de l'environnement.

"This will change my future with my family," the man said.

"Cela va changer mon avenir avec ma famille," a dit l'homme.

Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight $\alpha_{ij}$ of the annotation of the $j$-th source word for the $i$-th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b–d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.

One of the motivations behind the proposed approach was the use of a fixed-length context vector in the basic encoder–decoder approach. We conjectured that this limitation may make the basic encoder–decoder approach to underperform with long sentences. In Fig. 2, we see that the performance of RNNencdec dramatically drops as the length of the sentences increases. On the other hand, both RNNsearch-30 and RNNsearch-50 are more robust to the length of the sentences. RNNsearch-50, especially, shows no performance deterioration even with sentences of length 50 or more. This superiority of the proposed model over the basic encoder–decoder is further confirmed by the fact that the RNNsearch-30 even outperforms RNNencdec-50 (see Table 1).

(Bahdanau et al., 2014)
The end
References


