Effective Approaches to Attention-based Neural Machine Translation

Authors: Luong, Minh-Thang, Pham, Hieu, Manning, Christopher D
Conference: EMNLP 2015
Link: http://arxiv.org/abs/1508.04025

Effective Approaches to Attention-based Neural Machine Translation

I/p: source sentence $x = x_1, \ldots, x_n$

I/p: target sentence $y = y_1, \ldots, y_n$

NMT System: NN that directly models:

- the conditional probability $p(y|x)$

Components of NMT: Encoder

- Computes the representation s of each source sentence.

E.g. I need to look at whole sentence to translate a sentence sometimes.

He left : वह गया

He left his wallet : वह गया उसका बटुआ

वह अपना बटुआ छोड़ गया

Components of NMT: Decoder

- Generates one target word y_t at a time based on generated history $y_{<t}$ and representation s.
- Composes conditional probability as:

$$\log p(y|x) = \text{Sum}\{j=1,m\} \log p(y_j | y_{<j}, s)$$

- one can parameterize the probability of decoding each word y_j as:

$$p(y_j | y_{<j}, s) = \text{softmax}(g(h_j))$$

h_j hidden unit $h_j = f_{RNN}(h_{j-1}, s)$, $g(h_j)$ vocabulary sized vector

Components of NMT: Decoder

- Training objective:

$$J_t = \sum_{(x,y)\in\mathbb{D}} - \log p(y|x)$$

With $\mathbb{D}$ being our parallel training corpus

Attention-based Models

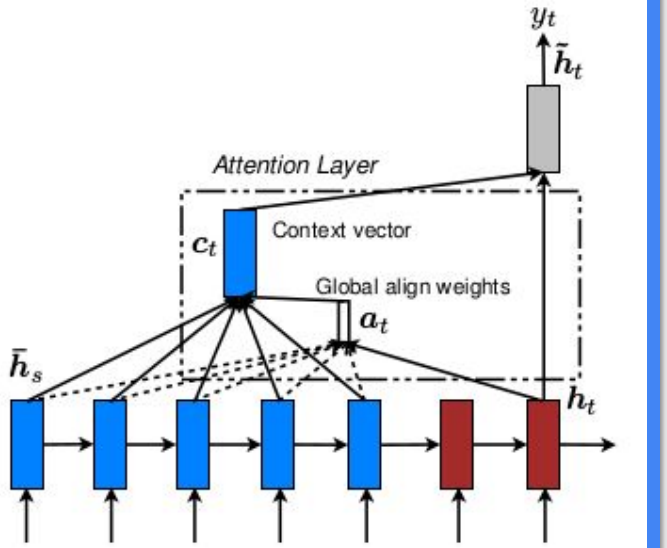

Figure 2: **Global attentional model** – at each time step t, the model infers a *variable-length* alignment weight vector $\boldsymbol{a}_t$ based on the current target state $\boldsymbol{h}_t$ and all source states $\bar{\boldsymbol{h}}_s$. A global context vector $\boldsymbol{c}_t$ is then computed as the weighted average, according to $\boldsymbol{a}_t$, over all the source states.

Concatenation layer employed to combine c_t and h_t

$$\tilde{h}_t = \tanh(W_c[c_t ; h_t])$$

The attentional vector $\tilde{h}_t$ is then fed through the softmax layer to produce the predictive distribution formulated as:

$$p(y_t|y_{<t}, x) = \text{softmax}(W_s \tilde{h}_t)$$

Attension-based Models

Types of attention based models:

1. Global
2. Local
3. Input-feeding approach

Global Attention

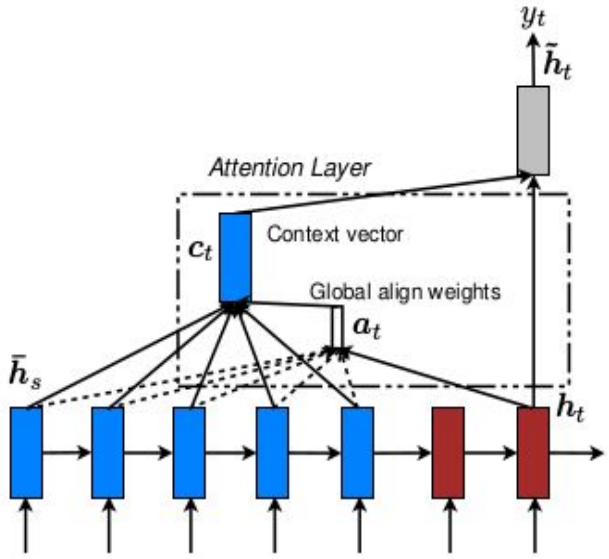

Figure 2: **Global attentional model** – at each time step t, the model infers a *variable-length* alignment weight vector $\boldsymbol{a}_t$ based on the current target state $\boldsymbol{h}_t$ and all source states $\bar{\boldsymbol{h}}_s$. A global context vector $\boldsymbol{c}_t$ is then computed as the weighted average, according to $\boldsymbol{a}_t$, over all the source states.

In this model type, a variable-length alignment vector a_t, whose size equals the number of time steps on the source side, is derived by comparing the current target hidden state *ht* with each source hidden state $\bar{h_s}$:

$$alpha\_t(s) = \text{align}(h_t, \bar{h_s})$$

$$= \frac{\exp(\text{score}(h_t, \bar{h_s}))}{\Sigma_{s'} \exp(\text{score}(h_t, \bar{h_{s'}}))} \qquad (7)$$

Global Attention

Here, score is referred as a content-based function for which we consider three different alternatives:

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W}_a \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W}_a [\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$

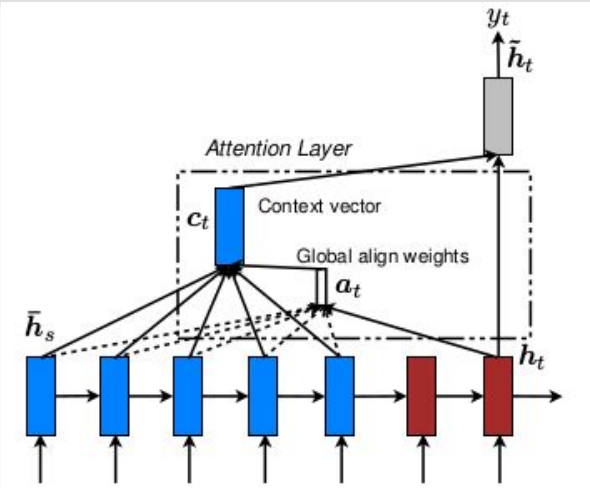

Figure 2: **Global attentional model** – at each time step t, the model infers a *variable-length* alignment weight vector $\boldsymbol{a}_t$ based on the current target state $\boldsymbol{h}_t$ and all source states $\bar{\boldsymbol{h}}_s$. A global context vector $\boldsymbol{c}_t$ is then computed as the weighted average, according to $\boldsymbol{a}_t$, over all the source states.

Global Attention

Drawback:
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.

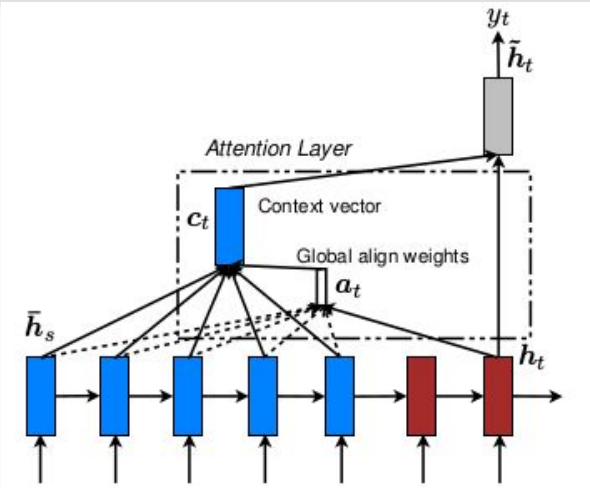

Figure 2: **Global attentional model** – at each time step t, the model infers a *variable-length* alignment weight vector a_t based on the current target state h_t and all source states $\bar{h}_s$. A global context vector c_t is then computed as the weighted average, according to a_t, over all the source states.

Local Attention

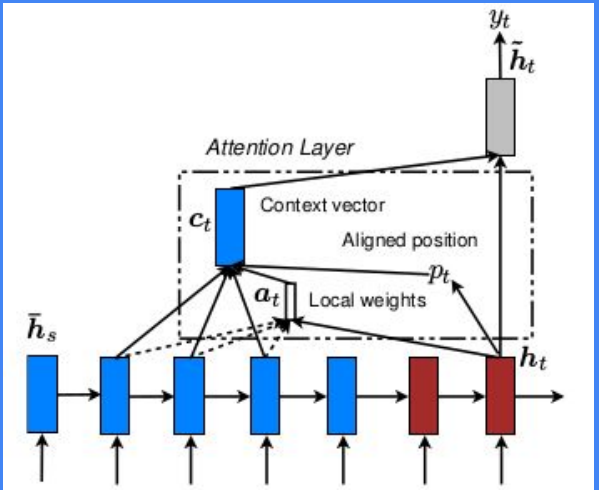

Figure 3: **Local attention model** – the model first predicts a single aligned position p_t for the current target word. A window centered around the source position p_t is then used to compute a context vector $\boldsymbol{c}_t$, a weighted average of the source hidden states in the window. The weights $\boldsymbol{a}_t$ are inferred from the current target state $\boldsymbol{h}_t$ and those source states $\bar{\boldsymbol{h}}_s$ in the window.

- This model takes inspiration from the tradeoff between the soft and hard attentional models proposed by Xu et al. (2015) to tackle the image caption generation task.

- This approach has an advantage of avoiding the expensive computation incurred in the soft attention and at the same time, is easier to train than the hard attention approach.

- In concrete details, the model first generates an aligned position p_t for each target word a_t time t.

Local Attention

- **Monotonic alignment (local-m)** – we simply set $p_t = t$ assuming that source and target sequences are roughly monotonically aligned. The alignment vector a t is defined according to Eq. (7).
- **Predictive alignment (local-p)** – instead of assuming monotonic alignments, our model predicts an aligned position as follows:

$$p_t = S \cdot \text{sigmoid}(v_p^\top \tanh(W_p h_t)), \quad (9)$$

W_p and v_p are the model parameters which will be learned to predict positions. S is the source sentence length. As a result of sigmoid, $p_t \in [0, S]$.

- To favor alignment points near p_t, we place a Gaussian distribution centered around p_t. Specifically, our alignment weights are now defined as:

$$\boldsymbol{a}_t(s) = \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \exp\left(-\frac{(s - p_t)^2}{2\sigma^2}\right) \quad (10)$$

Input-feeding Approach

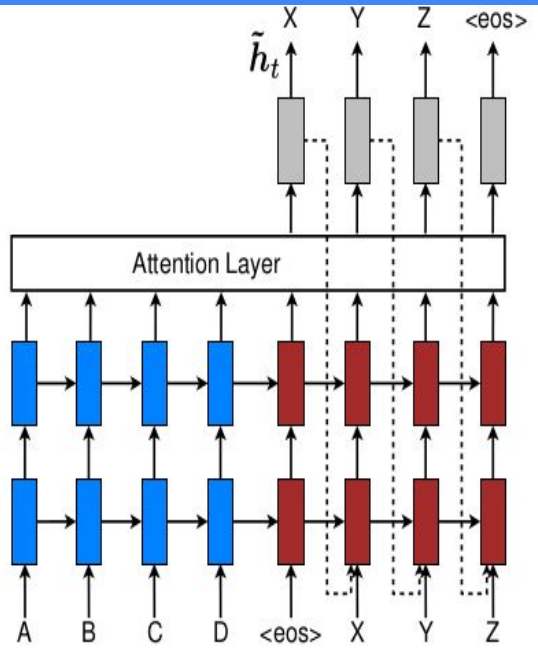

Figure 4: **Input-feeding approach** – Attentional vectors $\tilde{h}_t$ are fed as inputs to the next time steps to inform the model about past alignment decisions.

- In input-feeding approach, attentional vectors $\tilde{h}_t$ are concatenated with inputs at the next time steps as illustrated in figure

- The effects of having such connections are two-fold:
  (a) we hope to make the model fully aware of previous alignment choices
  (b) we create a very deep network spanning both horizontally and vertically.