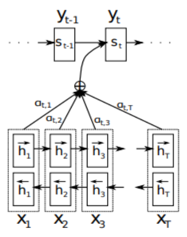
Ⅰ. Attention Mechanism:

1. **Bahdanau Attention:** *Neural Machine Translation by Learning to Jointly Align and Translate* (2014)



$$c_t = \sum_{j=1}^{T_x} a_{tj} h_j$$

$$a_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T_x} \exp(e_{tk})}$$

$$e_{tj} = V_a^T \tanh\left(W_a[s_{t-1}; h_j]\right)$$

where $h_t \in \mathbb{R}^n$ is a hidden state at time t, and c is a context vector generated from the sequence of the hidden states, e_{ij} is an alignment model which scores how well the inputs around position j and the output at position i match.

2. **Luong Attention:** *Effective Approaches to Attention-based Neural Machine Translation* (2015)

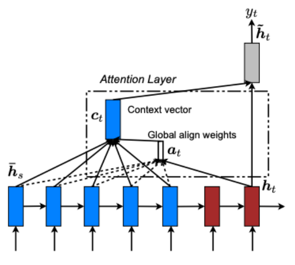


Figure 2: **Global attentional model** – at each time step t, the model infers a *variable-length* alignment weight vector a_t based on the current target state h_t and all source states $\bar{h}_s$. A global context vector c_t is then computed as the weighted average, according to a_t, over all the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

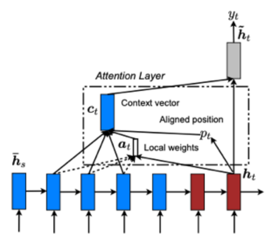


Figure 3: **Local attention model** – the model first predicts a single aligned position p_t for the current target word. A window centered around the source position p_t is then used to compute a context vector c_t, a weighted average of the source hidden states in the window. The weights a_t are inferred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

Global: $a_t(s) = align(h_t, \bar{h}_s)$

$$= \frac{\exp\left(score(h_t, \bar{h}_s)\right)}{\sum_{s'} \exp\left(score(h_t, \bar{h}_{s'})\right)}$$

$$score(h_t, \bar{h}_s) \begin{cases} h_t^\top, \bar{h}_s & dot \\ h_t^\top W_a \bar{h}_s & general \\ v_a^\top \tanh\left(W_a[h_t; \bar{h}_s]\right) & concat \end{cases}$$

$$a_t = softmax(W_a h_t)$$

Local: $a_t(s) = lign(h_t, \bar{h}_s) \exp\left(-\frac{(s-p_t)^2}{2\sigma^2}\right)$

$$p_t = S \cdot sigmoid\left(v_p^\top \tanh(W_p h_t)\right)$$

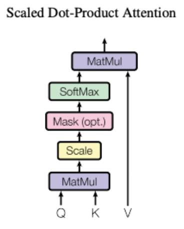
where $score$ is referred as a content-based function. W_p and v_p are the model parameters. S is the source sentence length.

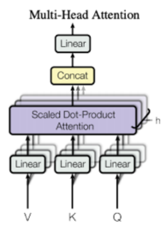
3. **Self-Attention & Multi-Head Attention:** *Attention Is All You Need* (2017)

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$head_i = Attention(q_i, K, V)$$

$$MultiHead(Q, K) = Concat(head_1, \cdots head_i)W^o$$





4. **Location Sensitive Attention:** *Attention-Based Models foe Speech Recognition* (2015)

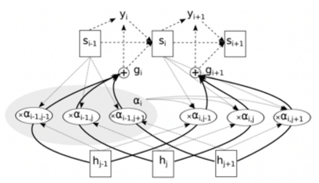


Figure 1: Two steps of the proposed attention-based recurrent sequence generator (ARSG) with a hybrid attention mechanism (computing α), based on both content (h) and location (previous α) information. The dotted lines correspond to Eq. (1), thick solid lines to Eq. (2) and dashed lines to Eqs. (3)–(4).

$$\alpha_i = Attend(s_{i-1}, \alpha_{i-1}, h)$$

$$g_i = \sum_{j=1}^{L} \alpha_{i,j} h_j$$

$$y_i \sim Generate(s_{i-1}, g_i)$$

$$s_i = Recurrency(s_{i-1}, g_i, y_i)$$

$$e_{i,j} = w^\top \tanh\left(Ws_{i-1} + Vh_j + b\right)$$

$$f_i = F * \alpha_{i-1}$$

$$e_{i,j} = w^\top \tanh\left(Ws_{i-1} + Vh_j + Uf_{i,j} + b\right)$$

$$\alpha_{i,j} = \exp(e_{i,j}) \Big/ \sum_{j=1}^{L} \exp(e_{i,j})$$

s_i is the current decoder hidden state and the bias value b is initialized to 0. The location feature f_i is convolved using the

5. **Hierarchical Attention:** *Hierarchical Attention Networks for Document Classification* (2016)

Word Attention:

$$u_{it} = \tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{\exp(u_{it}^{\mathrm{T}} u_w)}{\sum_t \exp(u_{it}^{\mathrm{T}} u_w)}$$

$$s_i = \sum_t \alpha_{it} h_{it}$$

Sentence Attention:

$$u_i = \tanh(W_s h_i + b_s)$$

$$\alpha_i = \frac{\exp(u_i^{\mathrm{T}} u_s)}{\sum_i \exp(u_i^{\mathrm{T}} u_s)}$$

$$v = \sum_i \alpha_i h_i$$

where h_{it} is word annotation, u_{it} is a hidden representation of h_{it}, u_w is a word level context vector, and α_{it} is a normalized importance weight.

6. **Dual Local and Global Attention:** *Interpretable Convolutional Neural Networks with Dual Local and Global Attention for Review Rating Prediction* (2017)

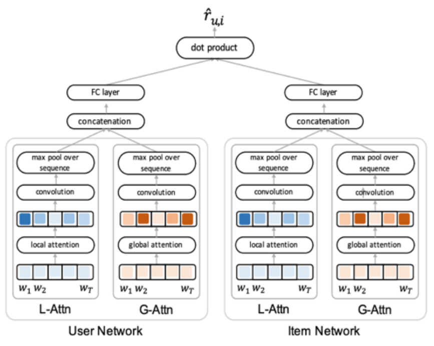


Figure 1: Architecture of D-Attn to extract latent representations of users and items. A user document D_u and an item document D_i are fed into (Left) the user network and (Right) the item network, respectively.

Local Attention(L-Attn):

$$\boldsymbol{X}_{l-att,i} = (\boldsymbol{x}_{i+\frac{-w+1}{2}}, \boldsymbol{x}_{i+\frac{-w+3}{2}}, \cdots \boldsymbol{x}_i, \cdots, \boldsymbol{x}_{i+\frac{w-1}{2}})^{\mathrm{T}}$$

$$s(i) = sigmoid\left(\boldsymbol{X}_{l-att,i} * \boldsymbol{W}_{l-att}^1 + b_{l-att}^1\right) \ i \in [1, T]$$

$$\hat{\boldsymbol{x}}_t^L = s(t)\boldsymbol{x}_t$$

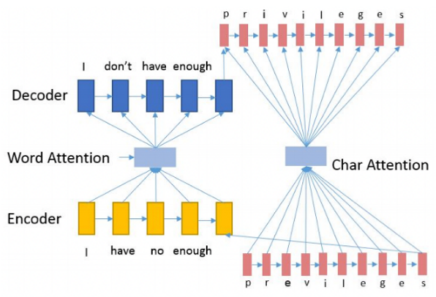
Global Attention(G-Attn):

$$\widehat{\boldsymbol{X}}_{g-att,i} = (\hat{\boldsymbol{x}}_i^G, \hat{\boldsymbol{x}}_{i+1}^G, \cdots \hat{\boldsymbol{x}}_{i+w_f-1}^G)^{\mathrm{T}}$$

$$\boldsymbol{Z}_{g-att}(i,j) = tanh(\widehat{\boldsymbol{X}}_{g-att,i} * \boldsymbol{W}_{g-att}(:,:,j) + b_{g-att}(j))$$

where x_i is center word, w is the kernel width, b is a bias vector, $*$ is an operation which means element wise multiplication and sum, W is a parameter matrix, w_f is the length of a filter.

7. **Nested Attention Hybrid Model:** *A Nested Attention Neural Hybrid Model for Grammatical Error Correction* (2017)



Global grammar and fluency correction with word-level attention, local spelling error correction with character-level attention.

$$c_s = \sum_{j=1}^{T} \alpha_{sj} h_j$$

$$\alpha_{sk} = \frac{u_{sk}}{\sum_{j=1}^{T} u_{sj}}, u_{sk} = \phi_1(d_s)^T \phi_2(h_k)$$

Here ϕ_1 and ϕ_2 denote feedforward linear transformations followed by a $tanh$ nonlinearity.

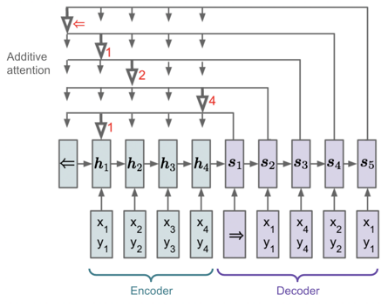
8. **Memory-based Attention:**

$$e_i = a(q, k_i) \quad address\ memory$$

$$\alpha_i = \frac{\exp(e_i)}{\sum_i \exp(e_i)}$$

$$c = \sum_i \alpha_i v_i$$

9. Pointer Network: *Pointer Networks* (2017)



$$u_j^i = v^{\top}\tanh(W_1 e_j + W_2 d_i)$$

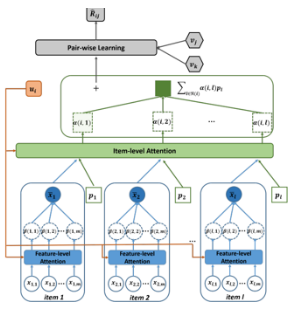
$$a_j^i = softmax(u_j^i)$$

$$d_i^{'} = \sum_{j=1}^{n} a_j^i e_j$$

where, e is encoder hidden states, d is decoder hidden states. *Ptr-Net* - An encoding RNN converts the input sequence to a code (blue) that is fed to the generating network (purple). At each step, the generating network produces a vector that modulates a content-based attention mechanism over inputs ([5, 2]). The output of the attention mechanism is a softmax distribution with dictionary size equal to the length of the input.

10. Knowledge-based Attention:

1) *Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention* (2017)



Item-level Attention:

$$a(i,l) = w_1^T \phi(W_{1u}u_i + W_{1v}v_l + W_{1p}p_l + W_{1x}\bar{x}_l + b_1) + c_1$$

$$\alpha(i,l) = \frac{\exp(a(i,l))}{\sum_{n\in\mathcal{R}(i)}\exp(a(i,n))}$$

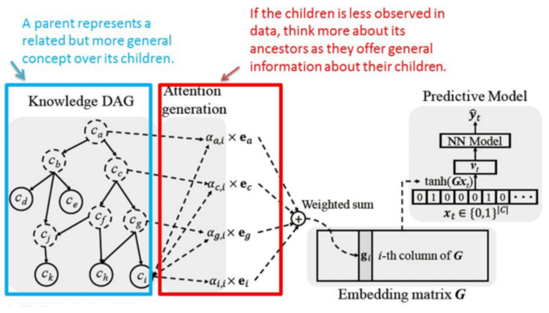
Component-level Attention:

$$b(i,l,m) = w_2^T \phi(W_{2u}u_i + W_{2x}x_{lm} + b_2) + c_2$$

$$\beta(i,l,m) = \frac{\exp(b(i,l,m))}{\sum_{n=1}^{|\{X_{l*}\}|}\exp(a(i,l,n))}$$

where, $\phi(x) = \max(0,x)$ is the ReLU function. $\bar{x}_l = \sum_{m=1}^{|\{X_{l*}\}|}\beta(i,l,m)\cdot X_{lm}$

2) *GRAM: Graph-based Attention Model For Healthcare Representation Learning* (2017)


A parent represents a related but more general concept over its children.
If the children is less observed in data, think more about its ancestors as they offer general information about their children.
Knowledge DAG
Attention generation
Predictive Model
NN Model
Weighted sum
$x_t \in \{0,1\}^{|C|}$
i-th column of G
Embedding matrix G

$$g_i = \sum_{j\in A(i)}\alpha_{ij}e_j, \sum_{j\in A(i)}\alpha_{ij} = 1$$

$$\alpha_{ij} = \frac{\exp(f(e_i,e_j))}{\sum_{k\in A(i)}\exp(f(e_i,e_k))}$$

$$f(e_i,e_j) = u_a^{\top}\tanh\left(W_a\begin{bmatrix}e_i\\ e_j\end{bmatrix} + b_a\right)$$

where $g_i \in \mathbb{R}^m$ denotes the final representation of the code, $A(i)$ is the indices of the code and ancestors of code, $W_a \in \mathbb{R}^{l\times 2m}$ is the weight matrix for the concatenation of e_i and e_j, and u_a is the weight vector for generation the scalar value.

11. Different ways of compute Attention Score:

- Content-based attention:

$$e_{ij} = score(s_{i-1}, h_j) = v_a^\top tanh(W_a s_{i-1} + U_a h_j)$$

- Location-based attention:

$$e_{ij} = score(a_{i-1}, h_j) = v_a^\top tanh(W h_j + U f_{ij})$$

- Hybrid attention:

$$e_{ij} = score(s_{i-1}, a_{i-1}, h_j) = v_a^\top tanh(W s_{i-1} + V h_j + U f_{ij} + b)$$

Name	**Attention Score Function**
Content-Base	$score(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathrm{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$
Additive(Concat)	$score(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{v}_a^\top \mathrm{tanh}\ (\boldsymbol{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$
Location-Base	$\alpha_{t,i} = softmax(\boldsymbol{W}_a \boldsymbol{s}_t)$
General	$score(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{W}_a \boldsymbol{h}_i$
Dot-Product	$score(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$
Scaled Dot-Product(^)	$score(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i / \sqrt{n}$

(^) It adds a scaling factor $1/\sqrt{n}$, motivated by the concern when the input is large, the softmax function may have an extremely small gradient, hard for efficient learning. $\boldsymbol{W}_a$ is a trainable weight matrix in the attention layer.

References: Attention? Attention![1]

[1] https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

II．可解释性：

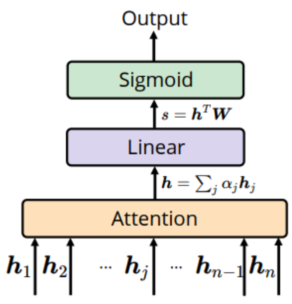
1. Understanding Attention for Text Classification

The *attention score* for the j-th token is (Token-level attention score):

$$a_j = \frac{h_j^\top V}{\lambda}$$

where V is the context vector. The corresponding *attention weight* would be:

$$\alpha_j = \frac{\exp{(a_j)}}{\sum_{j'} \exp{(a_{j'})}}$$



Output
Sigmoid
$s = \boldsymbol{h}^T \boldsymbol{W}$
Linear
$\boldsymbol{h} = \sum_j \alpha_j \boldsymbol{h}_j$
Attention
$\boldsymbol{h}_1$ $\boldsymbol{h}_2$ ··· $\boldsymbol{h}_j$ ··· $\boldsymbol{h}_{n-1}$ $\boldsymbol{h}_n$

Output (Instance-level polarity score): $s = h^\top W = \sum_j \alpha_j h_j^\top W = \sum_j \alpha_j$

Token-level polarity score: $s_j = h_j^\top W$

Divide the words into three types:

- **positive tokens**: tokens that frequently appear in positive training instances only,
- **negative tokens**: tokens that frequently appear in negative training instances only,
- **neutral tokens**: tokens that appear evenly across both positive and negative training instances

Analysis:

The update of s_e is $\frac{ds_e}{d\tau} = \frac{1}{m}(V^\top W/\lambda)\rho(e) + \frac{1}{m}\|W\|_2^2\pi(e) + \frac{1}{m}\sum_{(t,j)} y^{(t)}\beta^{(t)}\alpha_j^{(t)}e^\top e_j^{(t)}$

where $\rho(e) = \sum_{(t,j):e_j^{(t)}\equiv e} y^{(t)}\beta^{(t)}\alpha_j^{(t)}(s_e - s^{(t)})$

First, consider the case where λ is reasonably large, $A \approx 0$:

- **For positive tokens**, we have $B > 0$ and $C > 0$. The corresponding polarity scores will likely increase after each update when approaching the local minimum, and may end up with relatively large positive polarity scores eventually.
- **For negative tokens**, we have $B < 0$ and $C < 0$. The corresponding polarity scores will likely decrease after each update when approaching the local minimum, and may end up with relatively large negative polarity scores eventually.
- **For neutral tokens**, we have $B \approx 0$ and $C \approx 0$. Their polarity scores will likely not change significantly after each update when approaching the local minimum, and may end up with polarity scores that are neither significantly positive nor significantly negative eventually.

The update of a_e is $\frac{da_e}{d\tau} = \frac{1}{m\lambda^2}(V^\top W \cdot \lambda)\pi(e) + \frac{1}{m\lambda^2}\|V\|_2^2\rho(e) +$

$$\frac{1}{m\lambda^2}\sum_{(t,j)} y^{(t)}\beta^{(t)}\alpha_j^{(t)}e^\top e_j^{(t)}(s_j^{(t)} - s^{(t)})$$

- Term D. When $V^\top W > 0$, the positive tokens will receive a positive update whereas the negative tokens will receive a negative update from this term after each step. When $V^\top W < 0$, the influence is the other way around. It does not influence the attention scores of the neutral tokens much as the corresponding π(e) is approximately zero. When it is not close to zero, this term can lead to a gap between the final attention scores of the positive tokens and negative tokens.

- Terms E and F. Based on our analysis, $E > 0$, and $F \geq 0$ for polarity tokens, and $E \approx 0$ and $F \approx 0$ for neutral tokens. This means for the positive tokens and negative tokens; their attention scores will likely receive a positive value from this term after each update when approaching a local minimum. Their corresponding attention scores may end up with large positive scores eventually. For the neutral tokens, this term does not have much influence on their attention scores.

From here we can observe that when$V^\top W \cdot \lambda$ is small, the polarity tokens will likely end up with larger attention scores than the neutral tokens. This is actually a desirable situation – polarity tokens are likely more representative when used for predicting the underlying class labels, and therefor shall receive more "attention" in general.

2. Is Attention Interpretable?

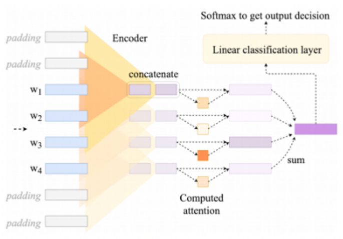


Figure 2: Flat attention network (FLAN) demonstrating a convolutional encoder. Each contextualized word representation is the concatenation of two sizes of convolutions: one applied over the input representation and its two neighbors to either side, and the other applied over the input representation and its single neighbor to either side. For details, see Appendix A.1.

$$u_i = \tanh(W_\ell h_i + b_\ell)$$

$$\alpha_i = \frac{\exp(u_i^T c_\ell)}{\sum_i \exp(u_i^T c_\ell)}$$

3. Attention is not not Explanation

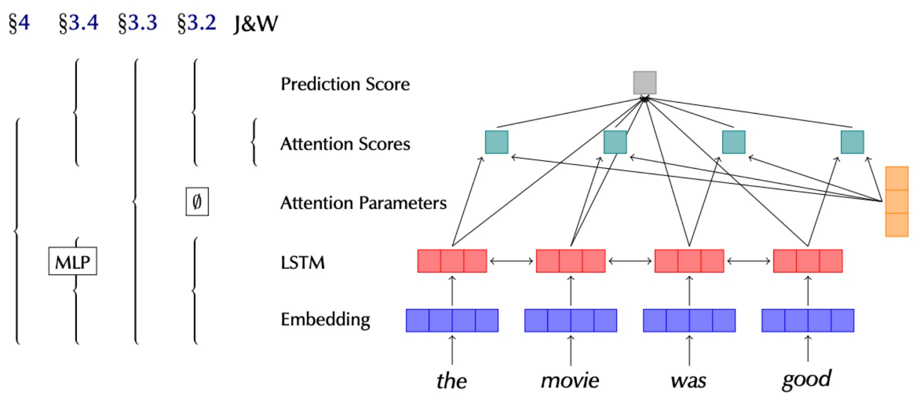


Figure 1: Schematic diagram of a classification LSTM model with attention, including the components manipulated or replaced in the experiments performed in Jain and Wallace (2019) and in this work (by section).

[2] https://plmsmile.github.io/2018/03/25/33-attention-summary/

[3] https://zhuanlan.zhihu.com/p/84490817