Attention on Attention for Image Captioning

Xavier Pleimling
Background and Motivation
Image Captioning

https://upload.wikimedia.org/wikipedia/commons/1/10/Tursiops_truncatus_01.jpg
Image Captioning

A bottlenose dolphin riding on an ocean wave

https://upload.wikimedia.org/wikipedia/commons/1/10/Tursiops_truncatus_01.jpg
The Attention Mechanism

Scaled Dot-Product Attention

Multi-Head Attention

The Attention Mechanism
Drawbacks

Not sure if attention result is related to RNN
Drawbacks

Not sure if attention result is related to

RNN

A seal riding on a cup of Sprite

???
Not sure if attention result is related to RNN.

**Causes**
1. Attention Model does not do well
2. Vectors have no good information

A seal riding on a cup of Sprite
Solution

Attention on Attention

Adding another attention to the existing attention

Solution

AoANet

Image

Encoder

CNN

Feature Vectors

RNN

Decoder

Text

A bottlenose dolphin riding on an ocean wave
Prior Work
Prior Work: Image Captioning

Early Approaches:


Prior Work: Image Captioning

Early Approaches:

More Recent Approaches:


Prior Work: Attention Mechanisms


Prior Work: Self-Attention

Prior Work: Self-Attention

Given $Q$ is a matrix of queries, $K$ is a matrix of keys, and $V$ is a matrix of values:

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

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O$$

where $\text{head}_i = \text{Attention}(QW^Q_i, KW^K_i, VW^V_i)$

Uses in the Transformer model:

1. “Encoder-decoder attention” layers
2. Self-attention layers for encoder
3. Self-attention layers for decoder
Prior Work: Self-Attention

Why Self-Attention?

<table>
<thead>
<tr>
<th>Layer Type</th>
<th>Complexity per Layer</th>
<th>Sequential Operations</th>
<th>Maximum Path Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(n^2 \cdot d)$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Recurrent</td>
<td>$O(n \cdot d^2)$</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>Convolutional</td>
<td>$O(k \cdot n \cdot d^2)$</td>
<td>$O(1)$</td>
<td>$O(\log_k(n))$</td>
</tr>
<tr>
<td>Self-Attention (restricted)</td>
<td>$O(r \cdot n \cdot d)$</td>
<td>$O(1)$</td>
<td>$O(n/r)$</td>
</tr>
</tbody>
</table>
Prior Work: Self-Attention

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
<th>Training Cost (FLOPs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EN-DE</td>
<td>EN-FR</td>
</tr>
<tr>
<td></td>
<td>EN-DE</td>
<td>EN-FR</td>
</tr>
<tr>
<td>ByteNet [15]</td>
<td>23.75</td>
<td>39.2</td>
</tr>
<tr>
<td>GNMT + RL [31]</td>
<td>25.16</td>
<td>40.46</td>
</tr>
<tr>
<td>ConvS2S [8]</td>
<td>26.03</td>
<td>40.56</td>
</tr>
<tr>
<td>Deep-Att + PosUnk Ensemble [32]</td>
<td></td>
<td>40.4</td>
</tr>
<tr>
<td>GNMT + RL Ensemble [31]</td>
<td>26.30</td>
<td>41.16</td>
</tr>
<tr>
<td>ConvS2S Ensemble [8]</td>
<td>26.36</td>
<td>41.29</td>
</tr>
<tr>
<td>Transformer (base model)</td>
<td>27.3</td>
<td>38.1</td>
</tr>
<tr>
<td>Transformer (big)</td>
<td><strong>28.4</strong></td>
<td><strong>41.0</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Self-attention can achieve state-of-the-art results in machine translation and computer vision

Prior Work: Attention Gates

Creation and application of attention gates are similar to:

**GLUs:**

**Multi-modal fusion:**

**GRUs/LSTMs:**


Prior Work

Overall, Attention on Attention is an extension of the existing attention mechanisms and can be applied to any of them.
Proposed Approach
Attention

Suppose the following:

\( Q \) is the set of queries, \( K \) is the set of keys, and \( V \) is the set of values

\[ f_{\text{sim}}(q_i, k_j) \]

is an arbitrary similarity model, with inputs \( q_i \) and \( k_j \), respectively, being the \( i \)th query in \( Q \) and \( j \)th key in \( K \)

\( v_j \) is the \( j \)th value in \( V \) corresponding to \( k_j \)

Then:

the attended vector \( \hat{v}_i \) for query \( q_i \) can be described as

\[ \hat{v}_i = \sum_j f_{\text{sim}}(q_i, k_j)v_j \]

This method will be denoted as \( f_{\text{att}}(Q,K,V) \)

\( f_{\text{att}}(Q,K,V) = \hat{V} \) with \( \hat{V} \) as the resulting weighted average vectors over \( V \)
Attention on Attention

Define $i$ to be the information vector and $g$ to be the attention gate

Given a query $q$ and the attention result $\hat{v}$ from $f_{\text{att}}(Q,K,V)$:

$$i = W^i_q q + W^i_v \hat{v} + b^i$$  and  $$g = \text{sigmoid}(W^g_q q + W^g_v \hat{v} + b^g)$$

where $W, b$ are associated linear projection constants

$g$ is then element-wise multiplied with $i$ to obtain attended information $\hat{i}$

$$AoA(f_{\text{att}}, Q, K, V) = \sigma(W^g_q Q + W^g_v f_{\text{att}}(Q, K, V) + b^g)$$
$$\odot (W^i_q Q + W^i_v f_{\text{att}}(Q, K, V) + b^i)$$ (6)
AoANet: Encoder (Refining Module)

\(A\) is a set of feature vectors from the CNN encoder network

\[A' = \text{LayerNorm}(A + \text{AoA}(\text{MultiHeadAttention}, W^Q_A, W^K_A, W^V_A))\]

Similar to the Transformer structure but with the feed-forward layer removed.

AoANet: Decoder

\[ c_t = AoA(MultiHeadAttention, W^Q h_t, W^K A, W^V A) \] with \( h_t \) being the LSTM output

\[ x_t = [W_e \Pi_t, \bar{a} + c_{t-1}] \]

\[ h_t, m_t = LSTM(x_t, h_{t-1}, m_{t-1}) \]

Loss and Optimization

Cross Entropy Loss:

\[ L_{XE}(\theta) = - \sum_{t=1}^{T} \log(p_{\theta}(y^*_t | y^*_{1:t-1})) \]

CIDEr-D Score Optimization:

\[ L_{RL}(\theta) = -E_{y_{1:T} \sim p_{\theta}}[r(y_{1:T})] \]
Implementation

**Encoder:** Faster-RCNN pre-trained on ImageNet and Visual Genome to retrieve 2048 dimensional vectors

**Decoder:** LSTM with hidden size 1024

**Training:**
- **Batch Size:** 10
- **Epochs:** 30 for $L_{XE}$ then 15 for $L_{RL}$
- **Learning Rate:** $2e^{-4}$ annealed by 0.8 every 3 epochs for $L_{XE}$, $2e^{-5}$ annealed by 0.5 if score does not improve for $L_{RL}$
- **Optimizer:** ADAM for $L_{XE}$, SCST for $L_{RL}$
Evaluation
Dataset and Metrics

Dataset:

MS COCO - 123,287 images with 5 captions each

“Kaparthy” split used for offline training

Metrics: BLEU, METEOR, ROUGE-L, CIDEr-D, SPICE

Quantitative Evaluation

Baselines:
LSTM, SCST, Up-Down, RFNet, GCN-LSTM, SGAE

All trained under XE loss and then optimized with RL loss

Offline Evaluation: Tested on the “Kaparthy” training split

Online Evaluation: Tested on the online COCO test server

Qualitative Evaluation also performed
## Offline Quantitative Evaluation

<table>
<thead>
<tr>
<th>Model</th>
<th>Cross-Entropy Loss</th>
<th>CIDEr-D Score Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B@1</td>
<td>B@4</td>
</tr>
<tr>
<td>Metric</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Single Model</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSTM [37]</td>
<td>-</td>
<td>29.6</td>
</tr>
<tr>
<td>SCST [31]</td>
<td>-</td>
<td>30.0</td>
</tr>
<tr>
<td>LSTM-A [50]</td>
<td>75.4</td>
<td>35.2</td>
</tr>
<tr>
<td>Up-Down [2]</td>
<td>77.2</td>
<td>36.2</td>
</tr>
<tr>
<td>RFNet [20]</td>
<td>76.4</td>
<td>35.8</td>
</tr>
<tr>
<td>GCN-LSTM [49]</td>
<td>77.3</td>
<td>36.8</td>
</tr>
<tr>
<td>SGAE [44]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AoANet (Ours)</td>
<td><strong>77.4</strong></td>
<td><strong>37.2</strong></td>
</tr>
<tr>
<td><strong>Ensemble/Fusion</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCST [31]</td>
<td>-</td>
<td>32.8</td>
</tr>
<tr>
<td>RFNet [20]</td>
<td>77.4</td>
<td>37.0</td>
</tr>
<tr>
<td>GCN-LSTM [49]</td>
<td>77.4</td>
<td>37.1</td>
</tr>
<tr>
<td>SGAE [44]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AoANet (Ours)</td>
<td><strong>78.7</strong></td>
<td><strong>38.1</strong></td>
</tr>
</tbody>
</table>

# Online Quantitative Evaluation

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
<th>METEOR</th>
<th>ROUGE-L</th>
<th>CIDEr-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCST [31]</td>
<td>78.1</td>
<td>93.7</td>
<td>61.9</td>
<td>86.0</td>
<td>47.0</td>
<td>75.9</td>
<td>35.2</td>
</tr>
<tr>
<td>LSTM-A [50]</td>
<td>78.7</td>
<td>93.7</td>
<td>62.7</td>
<td>86.7</td>
<td>47.6</td>
<td>76.5</td>
<td>35.6</td>
</tr>
<tr>
<td>Up-Down [2]</td>
<td>80.2</td>
<td>95.2</td>
<td>64.1</td>
<td>88.8</td>
<td>49.1</td>
<td>79.4</td>
<td>36.9</td>
</tr>
<tr>
<td>RFNet [20]</td>
<td>80.4</td>
<td>95.0</td>
<td>64.9</td>
<td>89.3</td>
<td>50.1</td>
<td>80.1</td>
<td>38.0</td>
</tr>
<tr>
<td>GCN-LSTM [49]</td>
<td>-</td>
<td>-</td>
<td>65.5</td>
<td>89.3</td>
<td>50.8</td>
<td>80.3</td>
<td>38.7</td>
</tr>
<tr>
<td>SGAE [44]</td>
<td>81.0</td>
<td>95.3</td>
<td>65.6</td>
<td>89.5</td>
<td>50.7</td>
<td>80.4</td>
<td>38.5</td>
</tr>
<tr>
<td>AoANet (Ours)</td>
<td><strong>81.0</strong></td>
<td><strong>95.0</strong></td>
<td><strong>65.8</strong></td>
<td><strong>89.6</strong></td>
<td><strong>51.4</strong></td>
<td><strong>81.3</strong></td>
<td><strong>39.4</strong></td>
</tr>
</tbody>
</table>
Qualitative Evaluation

Comparison Baseline:
Up-Down with the settings of AoANet

Observations:
1. AoANet counts objects of the same kind more accurately
2. AoANet properly determines the interactions of objects

Ablation Studies

Comparatively, AoA requires less computation than LSTM.

Ablation Studies

Human Evaluation

30 evaluators were invited to evaluate 100 images and asked to choose which of the two captions were better:

- Decoder with AoA: 49.2%
- Base: 21.2%
- Comparative: 29.7%
Generalization

MSR-VTT Dataset

1. A black and white horse runs around.
2. A horse galloping through an open field.
3. A horse is running around in green lush grass.
4. There is a horse running on the grassland.
5. A horse is riding in the grass.

<table>
<thead>
<tr>
<th></th>
<th>BLEU-4</th>
<th>CIDEr-D</th>
<th>ROUGE-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>33.53</td>
<td>38.83</td>
<td>56.90</td>
</tr>
<tr>
<td>decoder with AoA</td>
<td>37.22</td>
<td>42.44</td>
<td>58.32</td>
</tr>
</tbody>
</table>

https://production-media.paperswithcode.com/datasets/Screen_Shot_2021-01-28_at_9.51.08_PM.png
Strengths and Weaknesses
Strength #1 - Quite Efficient Compared To Normal LSTM

Less calculations are needed due to less hidden states
Strength #1 - Quite Efficient Compared To Normal LSTM

Less calculations are needed due to less hidden states

<table>
<thead>
<tr>
<th>Model</th>
<th>B@1</th>
<th>B@4</th>
<th>R</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>75.7</td>
<td>34.9</td>
<td>56.0</td>
<td>109.5</td>
</tr>
<tr>
<td>+ Enc: Refine (w/o AoA)</td>
<td>77.0</td>
<td>35.6</td>
<td>56.4</td>
<td>112.5</td>
</tr>
<tr>
<td>+ Enc: Refine (w/ AoA)</td>
<td>76.7</td>
<td>36.1</td>
<td>56.7</td>
<td>114.5</td>
</tr>
<tr>
<td>+ Dec: LSTM</td>
<td>76.8</td>
<td>35.9</td>
<td>56.6</td>
<td>113.5</td>
</tr>
<tr>
<td>+ Dec: AoA</td>
<td>76.6</td>
<td>35.8</td>
<td>56.6</td>
<td>113.8</td>
</tr>
<tr>
<td>+ Dec: LSTM + AoA</td>
<td></td>
<td></td>
<td></td>
<td>unstable training process</td>
</tr>
<tr>
<td>+ Dec: MH-Att</td>
<td>75.8</td>
<td>34.8</td>
<td>56.0</td>
<td>109.6</td>
</tr>
<tr>
<td>+ Dec: MH-Att, LSTM</td>
<td>76.6</td>
<td>35.8</td>
<td>56.7</td>
<td>113.8</td>
</tr>
<tr>
<td>+ Dec: MH-Att, AoA</td>
<td>76.9</td>
<td>36.1</td>
<td>56.6</td>
<td>114.3</td>
</tr>
<tr>
<td>Full: AoANet</td>
<td>77.4</td>
<td>37.2</td>
<td>57.5</td>
<td>119.8</td>
</tr>
</tbody>
</table>

Will help AoA stand out when performance is comparable

Strength #2 - Novel Compared To Previous Work

Very **unique and novel** concept

Strength #3 - Very Thorough Evaluation

- Multiple baselines and metrics
- Multiple angles of evaluation (quantitative, qualitative, ablations, etc.)
- High amount of evidence that AoA is effective:

Helps make a convincing argument for AoA being the new state-of-the-art for image captioning and perhaps other applications as well
Weakness #1 - Underdeveloped Training for AoANet

- Standard loss functions, especially XE loss
- Implementation details without much justification
- Why is CIDEr-D optimization needed?
- How XE and RL losses are handled together is not very clear
Weakness #2 - Lack of Accuracy Increase for Decoder

<table>
<thead>
<tr>
<th>Model</th>
<th>B@1</th>
<th>B@4</th>
<th>R</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>75.7</td>
<td>34.9</td>
<td>56.0</td>
<td>109.5</td>
</tr>
<tr>
<td>+ Enc: Refine (w/o AoA)</td>
<td>77.0</td>
<td>35.6</td>
<td>56.4</td>
<td>112.5</td>
</tr>
<tr>
<td>+ Enc: Refine (w/ AoA)</td>
<td>76.7</td>
<td>36.1</td>
<td>56.7</td>
<td>114.5</td>
</tr>
<tr>
<td>+ Dec: LSTM</td>
<td>76.8</td>
<td>35.9</td>
<td>56.6</td>
<td>113.5</td>
</tr>
<tr>
<td>+ Dec: AoA</td>
<td>76.6</td>
<td>35.8</td>
<td>56.6</td>
<td>113.8</td>
</tr>
<tr>
<td>+ Dec: LSTM + AoA</td>
<td>76.6</td>
<td>35.8</td>
<td>56.6</td>
<td>113.8</td>
</tr>
<tr>
<td>+ Dec: MH-Att</td>
<td>75.8</td>
<td>34.8</td>
<td>56.0</td>
<td>109.6</td>
</tr>
<tr>
<td>+ Dec: MH-Att, LSTM</td>
<td>76.6</td>
<td>35.8</td>
<td>56.7</td>
<td>113.8</td>
</tr>
<tr>
<td>+ Dec: MH-Att, AoA</td>
<td>76.9</td>
<td>36.1</td>
<td>56.6</td>
<td>114.3</td>
</tr>
<tr>
<td>Full: AoANet</td>
<td>77.4</td>
<td>37.2</td>
<td>57.5</td>
<td>119.8</td>
</tr>
</tbody>
</table>

Is it necessary to even have AoA when other attention methods can work just as well?

Weakness #3 - Methods/Evaluation Could Be Expanded?

Considering the AoA formula, try different attention mechanisms for $f_{att}$?

$$
\text{AoA}(f_{att}, Q, K, V) = \sigma(W^g_q Q + W^g_v f_{att}(Q, K, V) + b^g) \\
\odot (W^i_q Q + W^i_v f_{att}(Q, K, V) + b^i)
$$

Use different encoders other than Faster-RCNN?

Use different decoders other than LSTM?

Potential Ideas for Future Work
Future Work Ideas

- Bring AoA to other machine learning tasks, such as machine translation?
- Designing a better loss function?
- Designing a better decoder that utilizes AoA more effectively?
- Designing a better attention mechanism for the decoder step of image captioning?