CS11-747 Neural Networks for NLP

Attention

Graham Neubig

Carnegie Mellon University
Language Technologies Institute

Site
https://phontron.com/class/nn4nlp2021/
Encoder-decoder Models
(Sutskever et al. 2014)

Encoder

Decoder

I hate this movie

I hate this movie
Sentence Representations

Problem!

“You can’t cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!”
— Ray Mooney

• But what if we could use multiple vectors, based on the length of the sentence.

  this is an example →

  this is an example →
Attention
Basic Idea

(Bahdanau et al. 2015)

• Encode each word in the sentence into a vector

• When decoding, perform a linear combination of these vectors, weighted by “attention weights”

• Use this combination in picking the next word
Calculating Attention (1)

- Use “query” vector (decoder state) and “key” vectors (all encoder states)
- For each query-key pair, calculate weight
- Normalize to add to one using softmax

\[ a_1 = 2.1, \quad a_2 = -0.1, \quad a_3 = 0.3, \quad a_4 = -1.0 \]

\[ \alpha_1 = 0.76, \quad \alpha_2 = 0.08, \quad \alpha_3 = 0.13, \quad \alpha_4 = 0.03 \]
Calculating Attention (2)

• Combine together value vectors (usually encoder states, like key vectors) by taking the weighted sum

\[ kono \quad eiga \quad ga \quad kirai \]

Value Vectors

\[ \alpha_1 = 0.76 \quad \alpha_2 = 0.08 \quad \alpha_3 = 0.13 \quad \alpha_4 = 0.03 \]

• Use this in any part of the model you like
A Graphical Example

Image from Bahdanau et al. (2015)
Attention is not Alignment!
(Koehn and Knowles 2017)

- Attention is often blurred
- Attention is often off by one
- It can even be manipulated to be non-intuitive! (Jain and Wallace 2019, Pruthi et al. 2020)
Attention Score Functions (1)

• $q$ is the query and $k$ is the key

• **Multi-layer Perceptron** (Bahdanau et al. 2015)

  $$a(q, k) = w_2^\top \tanh(W_1 [q; k])$$

  • Flexible, often very good with large data

• **Bilinear** (Luong et al. 2015)

  $$a(q, k) = q^\top W k$$
Attention Score Functions (2)

• **Dot Product** (Luong et al. 2015)
  \[ a(q, k) = q^T k \]
  
  • No parameters! But requires sizes to be the same.

• **Scaled Dot Product** (Vaswani et al. 2017)
  
  • *Problem*: scale of dot product increases as dimensions get larger
  
  • *Fix*: scale by size of the vector
  \[ a(q, k) = \frac{q^T k}{\sqrt{|k|}} \]
Let’s Try it Out!
batched_attention.py

Try it Yourself: This code uses MLP attention. What would you do to implement a different variety of attention?
Self Attention
(Cheng et al. 2016, Vaswani et al. 2017)

• Each element in the sentence attends to other elements → context sensitive encodings!

```
this  is  an  example

this  |  is  |  an  | example
|      |     |      |      
|      |     |      |      
|      |     |      |      
|      |     |      |      |

|      |     |      |      |
|      |     |      |      |
```
**Multi-headed Attention**

- **Idea:** multiple attention “heads” focus on different parts of the sentence

- e.g. Different heads for “copy” vs regular (Allamanis et al. 2016)

- Or multiple independently learned heads (Vaswani et al. 2017)

- Or one head for every hidden node! (Choi et al. 2018)
An Interesting Case Study: “Attention is All You Need”
(Vaswani et al. 2017)
Summary of the "Transformer" (Vaswani et al. 2017)

- A sequence-to-sequence model based entirely on attention
- Strong results on translation, a wide variety of other tasks
- Fast: only matrix multiplications
Attention Tricks

• **Self Attention:** Each layer combines words with others

• **Multi-headed Attention:** 8 attention heads learned independently

• **Normalized Dot-product Attention:** Remove bias in dot product when using large networks

• **Positional Encodings:** Make sure that even if we don’t have RNN, can still distinguish positions
Training Tricks

• **Layer Normalization**: Help ensure that layers remain in reasonable range

• **Specialized Training Schedule**: Adjust default learning rate of the Adam optimizer

• **Label Smoothing**: Insert some uncertainty in the training process

• **Masking for Efficient Training**
Masking for Training

- We want to perform training in as few operations as possible using big matrix multiplies.

- We can do so by “masking” the results for the output.
A Caveat: Attention Is Not All You Need?

• Transformers are very popular, for good reason, but

• They can be slow to decode at test time (Zhang et al. 2018)

• They don't necessarily outperform RNNs on the decoder side of seq2seq tasks (Chen et al. 2018)

• They can be hard to train on small data (Nguyen and Salazar 2019)

• Use them, but also be aware of limitations!
Better Modeling for Attention
Incorporating Markov Properties
(Cohn et al. 2015)

• **Intuition:** attention from last time tends to be correlated with attention this time

• Add information about the last attention when making the next decision
Hard Attention

• Instead of a soft interpolation, make a **zero-one decision** about where to attend (Xu et al. 2015)

  • Harder to train, requires methods such as reinforcement learning (see later classes)

• Perhaps this helps interpretability? (Lei et al. 2016)
Monotonic Attention
(e.g. Yu et al. 2016)

• In some cases, we might know the output will be the same order as the input
  • Speech recognition, incremental translation, morphological inflection (?), summarization (?)

• **Basic idea:** hard decisions about whether to read more
Better Training for Attention
Coverage

• **Problem:** Neural models tends to drop or repeat content

• **Solution:** Model how many times words have been covered
  
  • Impose a penalty if attention not approx. 1 over each word (Cohn et al. 2015)
  
  • Add embeddings indicating coverage (Mi et al. 2016)
Bidirectional Training
(Cohn et al. 2015)

- **Intuition:** Our attention should be roughly similar in forward and backward directions

- **Method:** Train so that we get a bonus based on the trace of the matrix product for training in both directions

\[ \text{tr}(A_{X \rightarrow Y} A_{Y \rightarrow X}^T) \]
Supervised Training
(Mi et al. 2016)

- Sometimes we can get “gold standard” alignments \textit{a-priori}
  
  - Manual alignments
  
  - Pre-trained with strong alignment model

- \textbf{Train the model to match} these strong alignments
What Else Can We Attend To?
Input Sentence: Copy

- Like the previous explanation
- But also, more directly through a *copy mechanism* (Gu et al. 2016)
• If you have a translation dictionary, use it to bias outputs (Arthur et al. 2016)

<table>
<thead>
<tr>
<th>Attention</th>
<th>I</th>
<th>come</th>
<th>from Tunisia</th>
</tr>
</thead>
<tbody>
<tr>
<td>watashi</td>
<td>0.6</td>
<td>0.03</td>
<td>0.01</td>
</tr>
<tr>
<td>ore</td>
<td>0.2</td>
<td>0.01</td>
<td>0.02</td>
</tr>
<tr>
<td>kuru</td>
<td>0.01</td>
<td>0.3</td>
<td>0.01</td>
</tr>
<tr>
<td>kara</td>
<td>0.02</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>chunijia</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>oranda</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Sentence-level dictionary probability matrix

Dictionary probability for current word
Previously Generated Things

• In language modeling, attend to the previous words (Merity et al. 2016)

\[ p(Yellen) = g \cdot p_{vocabulary}(Yellen) + (1 - g) \cdot p_{ptr}(Yellen) \]

• In translation, attend to either input or previous output (Vaswani et al. 2017)
Various Modalities

- Images (Xu et al. 2015)

- Speech (Chan et al. 2015)
Hierarchical Structures
(Yang et al. 2016)

- Encode with attention over each sentence, then attention over each sentence in the document
Multiple Sources

- Attend to multiple sentences (Zoph et al. 2015)
  
  **Source 1:** UNK Aspekte sind ebenfalls wichtig.
  
  **Target:** UNK aspects are important, too.
  
  **Source 2:** Les aspects UNK sont également importants.

- Libovicky and Helcl (2017) compare multiple strategies

- Attend to a sentence and an image (Huang et al. 2016)
Questions?