TTIC 31210: Advanced Natural Language Processing

Kevin Gimpel
Spring 2019

Lecture 6: Finish Transformers; Sequence-to-Sequence Modeling and Attention
Roadmap

• intro (1 lecture)
• deep learning for NLP (5 lectures)
• structured prediction: sequence labeling, syntactic and semantic parsing, dynamic programming (4 lectures)
• generative models, latent variables, unsupervised learning, variational autoencoders (2 lectures)
• Bayesian methods in NLP (2 lectures)
• Bayesian nonparametrics in NLP (2 lectures)
• review & other topics (1 lecture)
Assignments

• Any last-minute questions about Assignment 1?

• Assignment 2 will be posted tonight, due in 2 weeks
Today

- finish transformers
- sequence-to-sequence modeling and attention
Transformer

- effective encoder for text sequences (and other data)
- no recurrent/convolutional modules
- only attention (various forms)

Vaswani et al. (2017): Attention Is All You Need
Attention

• attention is a useful generic tool
• often used to replace a sum or average with an attention-weighted sum
Attention

• e.g., for a word averaging encoder:

\[ f_{\text{avg}}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \text{emb}(x_i) \]

\[ f_{\text{att}}(\mathbf{x}) = \sum_{i=1}^{n} \text{att}(x_i, i, \mathbf{x}) \text{emb}(x_i) \]

“attention” function, returns a scalar
Example Attention Function

\[ f_{\text{att}}(x) = \sum_{i=1}^{n} \text{att}(x_i, i, x) \text{emb}(x_i) \]

\[ \text{att}(x_i, i, x) \propto \exp\{w^\top \text{emb}(x_i)\} \]

• introduces a new parameter vector \( w \) which is learned along with the word embeddings
• attention is normalized over the sentence length
Queries, Keys, and Values

• we can often think of attention functions in terms of these abstractions
• query = what you use to search
• key = the field that you’re comparing to
• value = the field that you return

Vaswani et al. (2017): Attention Is All You Need
• considering attention as query/key/value suggests using different spaces for different roles
• e.g., we could use separate transformations of the embedding space for keys and values:

\[ f_{\text{att}}(\mathbf{x}) = \sum_{i=1}^{n} \text{att}(x_i, i, \mathbf{x}) \left( \mathbf{W}^{(v)} \text{emb}(x_i) \right) \]

\[ \text{att}(x_i, i, \mathbf{x}) \propto \exp \left\{ \mathbf{w}^\top \left( \mathbf{W}^{(k)} \text{emb}(x_i) \right) \right\} \]
Multi-Head Attention

• we may want to learn multiple attention functions in parallel
• why? so that they can learn complementary functionality for the task

Vaswani et al. (2017): Attention Is All You Need
Multi-Head Attention

- in the transformer, each attention head uses projections to lower dimension, followed by concatenation of the outputs from each head

Vaswani et al. (2017): Attention Is All You Need
Transformer

Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Vaswani et al. (2017): Attention Is All You Need
Self-Attention

- many possibilities for self-attention functions
- intuitively, the following weights a word based on how similar it is to all other words in the sequence:

$$ \text{att}(x_i, i, \mathbf{x}) \propto \exp \left\{ \sum_{j=1}^{n} \text{emb}(x_i)^\top \text{emb}(x_j) \right\} $$

- can be combined with query/key/value-specific transformations and multiple heads

Vaswani et al. (2017): Attention Is All You Need
Word Position Information

- transformer adds embedding of position to word embedding in input
- predetermined/fixed sinusoidal embeddings or learned positional embeddings

Vaswani et al. (2017): *Attention Is All You Need*
Transformer

- developed for a setting with both encoding and decoding
- transformer decoder uses similar components to encoder, along with attention on encoder output

Vaswani et al. (2017): Attention Is All You Need
Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb ‘making’, completing the phrase ‘making...more difficult’. Attendations here shown only for the word ‘making’. Different colors represent different heads. Best viewed in color.
What do Transformers Learn?
What do Transformers Learn?

Figure 5: Many of the attention heads exhibit behaviour that seems related to the structure of the sentence. We give two such examples above, from two different heads from the encoder self-attention at layer 5 of 6. The heads clearly learned to perform different tasks.
Applications

• transformer encoders have become widely used in NLP
• developed for sequence transduction tasks (e.g., machine translation)
• also highly effective for language modeling, parsing, etc.
• we’ll briefly discuss two pretrained models based on transformers
Generative Pre-Training with Transformer Language Models

Radford et al. (2018): Improving language understanding with unsupervised learning
OpenAI GPT

• train a transformer language model (predict the next word)
• training data = passages from books
• applied to several NLP tasks that involve predicting things about sentences or filling in the next word
• transformer architecture (likely?) helps with capturing long-distance dependencies
• they then scaled it up to produce GPT-2
Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

GPT-2: The scientist named the population, after their distinctive horn, Ovid’s Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. “By the time we reached the top of one peak, the water looked blue, with some crystals on top,” said Pérez.
<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BERT (ensemble)</td>
<td>87.433</td>
<td>93.160</td>
</tr>
<tr>
<td></td>
<td>Google AI Language</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><a href="https://arxiv.org/abs/1810.04805">https://arxiv.org/abs/1810.04805</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>nlnet (ensemble)</td>
<td>85.356</td>
<td>91.202</td>
</tr>
<tr>
<td></td>
<td>Microsoft Research Asia</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>QANet (ensemble)</td>
<td>84.454</td>
<td>90.490</td>
</tr>
<tr>
<td></td>
<td>Google Brain &amp; CMU</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stanford University (Rajpurkar et al. '16)</td>
<td>82.304</td>
<td>91.221</td>
</tr>
</tbody>
</table>
SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph. How will your system compare to humans on this task?

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BERT + DAE + AoA (ensemble)</td>
<td>87.147</td>
<td>89.474</td>
</tr>
<tr>
<td></td>
<td>Joint Laboratory of HIT and iFLYTEK Research</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Rajpurkar &amp; Jia et al. '18)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BERT + ConvLSTM + MTL + Verifier (ensemble)</td>
<td>86.730</td>
<td>89.286</td>
</tr>
<tr>
<td></td>
<td>Layer 6 AI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>BERT + N-Gram Masking + Synthetic Self-Training (ensemble)</td>
<td>86.673</td>
<td>89.147</td>
</tr>
<tr>
<td></td>
<td>Google AI Language</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><a href="https://github.com/google-research/bert">GitHub link</a></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# General Language Understanding Evaluation (GLUE) Leaderboard

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Score</th>
<th>CoLA</th>
<th>SST-2</th>
<th>MRPC</th>
<th>STS-B</th>
<th>QQP</th>
<th>MNLI-m</th>
<th>QNLI</th>
<th>RTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BERT: 24-layers, 1024-hidden, 16-heads</td>
<td>80.4</td>
<td>60.5</td>
<td>94.9</td>
<td>85.4/89.3</td>
<td>87.6/86.5</td>
<td>89.3/72.1</td>
<td>86.7</td>
<td>91.1</td>
<td>70.1</td>
</tr>
<tr>
<td>2</td>
<td>SingleTask Pretrain Transformer</td>
<td>72.8</td>
<td>45.4</td>
<td>91.3</td>
<td>75.7/82.3</td>
<td>82.0/80.0</td>
<td>88.5/70.3</td>
<td>82.1</td>
<td>88.1</td>
<td>56.0</td>
</tr>
<tr>
<td>3</td>
<td>BiLSTM+ELMo+Attn</td>
<td>70.5</td>
<td>36.0</td>
<td>90.4</td>
<td>77.9/84.9</td>
<td>75.1/73.3</td>
<td>84.7/64.8</td>
<td>76.4</td>
<td>79.9</td>
<td>56.8</td>
</tr>
<tr>
<td>Rank</td>
<td>Name</td>
<td>Model</td>
<td>URL</td>
<td>Score</td>
<td>CoLA</td>
<td>SST-2</td>
<td>MRPC</td>
<td>STS-B</td>
<td>QQP</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------</td>
<td>--------------------------------------------</td>
<td>-----</td>
<td>-------</td>
<td>------</td>
<td>-------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>GLUE Human Baselines</td>
<td>GLUE Human Baselines</td>
<td></td>
<td>87.1</td>
<td>66.4</td>
<td>97.8</td>
<td>86.3/80.8</td>
<td>92.7/92.6</td>
<td>59.5/80.4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Microsoft D365 AI &amp; MSR AMT-DNN++ (BigBird)</td>
<td></td>
<td></td>
<td>83.8</td>
<td>65.4</td>
<td>95.6</td>
<td>91.1/88.2</td>
<td>89.6/89.0</td>
<td>72.7/89.6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>王玮</td>
<td>ALICE large (Alibaba DAMO NLP)</td>
<td></td>
<td>83.3</td>
<td>63.5</td>
<td>95.2</td>
<td>91.8/89.0</td>
<td>89.8/88.8</td>
<td>74.0/90.4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Stanford Hazy Research</td>
<td>Snorkel MeTaL</td>
<td></td>
<td>83.2</td>
<td>63.8</td>
<td>96.2</td>
<td>91.5/88.5</td>
<td>90.1/89.7</td>
<td>73.1/89.9</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>张倬胜</td>
<td>SemBERT</td>
<td></td>
<td>82.9</td>
<td>62.3</td>
<td>94.6</td>
<td>91.2/88.3</td>
<td>87.8/86.7</td>
<td>72.8/89.8</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Anonymous Anonymous</td>
<td>BERT + BAM</td>
<td></td>
<td>82.3</td>
<td>61.5</td>
<td>95.2</td>
<td>91.3/88.3</td>
<td>88.6/87.9</td>
<td>72.5/89.7</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Jason Phang</td>
<td>BERT on STILTs</td>
<td></td>
<td>82.0</td>
<td>62.1</td>
<td>94.3</td>
<td>90.2/86.6</td>
<td>88.7/88.3</td>
<td>71.9/89.4</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Jacob Devlin</td>
<td>BERT: 24-layers, 16-heads, 1024-</td>
<td></td>
<td>80.5</td>
<td>60.5</td>
<td>94.9</td>
<td>89.3/85.4</td>
<td>87.6/86.5</td>
<td>72.1/89.3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Neil Houlsby</td>
<td>BERT + Single-task Adapters</td>
<td></td>
<td>80.2</td>
<td>59.2</td>
<td>94.3</td>
<td>88.7/84.3</td>
<td>87.3/86.1</td>
<td>71.5/89.4</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Alec Radford</td>
<td>Single-task Pretrain Transformer</td>
<td></td>
<td>72.8</td>
<td>45.4</td>
<td>91.3</td>
<td>82.3/75.7</td>
<td>82.0/80.0</td>
<td>70.3/88.5</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>GLUE Baselines</td>
<td>BiLSTM+ELMo+Attn</td>
<td></td>
<td>70.0</td>
<td>33.6</td>
<td>90.4</td>
<td>84.4/78.0</td>
<td>74.2/72.3</td>
<td>63.1/84.3</td>
<td></td>
</tr>
</tbody>
</table>
BERT Overview

- transformer encoder with 24 layers, hidden size 1024, 16 attention heads; 340M parameters
- learned positional embeddings (up to 512)
- word prediction softmax layers are attached to the output layer (at every position)
- two training tasks:
  - masked LM: mask certain input words, predict them
  - next-sentence classification: encode two sentences, classify whether second follows first in a corpus

Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

- input uses WordPieces (vocabulary of size 30,000)
- first token is [CLS]
- special token [SEP] separates sentences

Pretraining Task 1: Masked Word Prediction

The training data generator chooses 15% of tokens at random, e.g., in the sentence "my dog is hairy" it chooses the word "hairy." It then performs the following procedure:

- Rather than *always* replacing the chosen words with [MASK], the data generator will do the following:
  - 80% of the time: Replace the word with the [MASK] token, e.g., "my dog is hairy" → "my dog is [MASK]"
  - 10% of the time: Replace the word with a random word, e.g., "my dog is hairy" → "my dog is apple"
  - 10% of the time: Keep the word unchanged, e.g., "my dog is hairy" → "my dog is hairy." The purpose of this is to bias the representation towards the actual observed word.

Pretraining Task 2: Next Sentence Classification

Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]

penguin [MASK] are flight #less birds [SEP]

Label = NotNext

Today

• finish transformers
• sequence-to-sequence modeling and attention
Sequence-to-Sequence Modeling

• data: <input sequence, output sequence> pairs
• use one RNN to encode input sequence
• use another RNN to decode the output sequence from the input sequence vector
• often more generally called “encoder-decoder” models
Application: Machine Translation

- encode source sentence into a vector, decode translation

Kalchbrenner & Blunsom (2013): *Recurrent Continuous Translation Models*


Sutskever et al. (2014): *Sequence to Sequence Learning with Neural Networks*
Application: Chatbots

**Human:** what is the purpose of life?
**Machine:** to serve the greater good.
**Human:** what is the purpose of living?
**Machine:** to live forever.
**Human:** what is the purpose of existence?
**Machine:** to find out what happens when we get to the planet earth.
**Human:** where are you now?
**Machine:** i’m in the middle of nowhere.
**Human:** what is the purpose of dying?
**Machine:** to have a life.
**Human:** what is the purpose of being intelligent?
**Machine:** to find out what it is.
**Human:** what is the purpose of emotions?
**Machine:** i don’t know.

Vinyals & Le (2015): A Neural Conversational Model
Application: Parsing

Figure 2: Example parsing task and its linearization.

{Vinyals, Kaiser} et al. (2015): Grammar as a Foreign Language
RNN Autoencoders

• encode sentence, decode sentence
• can use final hidden vector of sentence encoder as a sentence embedding

Li et al. (2015): A Hierarchical Neural Autoencoder for Paragraphs and Documents
Skip-Thoughts

- encode sentence using an RNN
- decode two neighboring sentences
- use different RNNs for previous and next sentences
- also pass encoder sentence vector on each decoding step

...I got back home  I could see the cat on the steps  This was strange ...

Kiros et al. (2015): *Skip-Thought Vectors*
query sentence:

*im sure youll have a glamorous evening , she said , giving an exaggerated wink .*

nearest neighbor:

*im really glad you came to the party tonight , he said , turning to her .*

Kiros, Zhu, Salakhutdinov, Zemel, Torralba, Urtasun, Fidler (2015)
Encoder-Decoder Models for Neural Machine Translation

- encode source sentence into a vector, decode translation

Kalchbrenner & Blunsom (2013): *Recurrent Continuous Translation Models*


Sutskever et al. (2014): *Sequence to Sequence Learning with Neural Networks*
Encoder-Decoder Models for Neural Machine Translation

Figure 3: A graphical depiction of the two RCTMs. Arrows represent full matrix transformations while lines are vector transformations corresponding to columns of weight matrices.

Kalchbrenner & Blunsom (2013): Recurrent Continuous Translation Models
Encoder-Decoder Models for Neural Machine Translation

- encode source sentence into a vector, decode translation

Figure 1: An illustration of the proposed RNN Encoder–Decoder.

Encoder-Decoder Models for Neural Machine Translation

- encode source sentence into a vector, decode translation

Sutskever et al. (2014): *Sequence to Sequence Learning with Neural Networks*
Encoder as a Sentence Embedding Model?

Sutskever et al. (2014): *Sequence to Sequence Learning with Neural Networks*
Encoder as a Sentence Embedding Model?

Sutskever et al. (2014): *Sequence to Sequence Learning with Neural Networks*
Sequence-to-Sequence Model

encoding input sequence:

\[ h_u = \tanh \left( W^{(x)} x_u + W^{(h)} h_{u-1} + b^{(h)} \right) \]
Sequence-to-Sequence Model

**encoder:** $h_u = \tanh \left( W^{(x)} x_u + W^{(h)} h_{u-1} + b^{(h)} \right)$

**decoder:** $s_t = \tanh \left( W^{(y)} \text{emb}(y_{t-1}) + W^{(s)} s_{t-1} + b^{(s)} \right)$

$y_t = \arg\max_{y \in \mathcal{O}} s_t^\top \text{emb}(y)$

$s_0 = h_{|x|}$

![Diagram](attachment:image.png)
Extension: Attention

Neural Machine Translation by Jointly Learning to Align and Translate

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho  Yoshua Bengio*
Université de Montréal
Figure 1: The graphical illustration of the proposed model trying to generate the $t$-th target word $y_t$ given a source sentence $(x_1, x_2, \ldots, x_T)$.

Bahdanau et al. (2015): Neural Machine Translation by Jointly Learning to Align and Translate
Extension: Attention

Bahdanau et al. (2015): Neural Machine Translation by Jointly Learning to Align and Translate
• disclaimer: the version I will present is a little simpler than the Bahdanau et al version but retains the key ideas
• see the paper for more details
Adding Attention

$$\alpha_{t,u} \propto \exp\{att(s_{t-1}, h_u)\}$$

$att$ function models association between all pairs of hidden vectors in encoder and decoder.
Adding Attention

$$\alpha_{t,u} \propto \exp\{att(s_{t-1}, h_u)\}$$

How is the $att$ function defined?

Bahdanau et al. used a feed-forward network:

$$att(s_{t-1}, h_u) = v_a^\top \tanh \left( W^{(as)} s_{t-1} + W^{(ah)} h_u \right)$$
Adding Attention

\[ \alpha_{t,u} \propto \exp\{att(s_{t-1}, h_u)\} \]

create new decoder hidden vector based on attention-weighted sum of encoder hidden vectors:

\[ c_t = \sum_{u=1}^{|x|} \alpha_{t,u} h_u \]

![Diagram showing attention mechanism](image)
$\alpha_{t,u} \propto \exp\{\text{att}(s_{t-1}, h_u)\}$

$|x|$

$c_t = \sum_{u=1}^{u} \alpha_{t,u} h_u$

create new decoder hidden vector based on attention-weighted sum of encoder hidden vectors:
\[ c_t = \sum_{u=1}^{\left| \text{x} \right|} \alpha_{t,u} h_u \]

new RNN decoder hidden state update equation:

\[ s_t = \tanh \left( W^{(y)} \text{emb}(y_{t-1}) + W^{(s)} s_{t-1} + W^{(c)} c_t + b^{(s)} \right) \]
\[ s_t = \tanh \left( W^{(y)} \text{emb}(y_{t-1}) + W^{(s)} s_{t-1} + W^{(c)} c_t + b^{(s)} \right) \]

new prediction equation:

\[ y_t = \arg\max_{y \in \mathcal{O}} \text{score}(\text{emb}(y), s_t, c_t) \]
Notes

• while initially developed for machine translation, attention is useful for many other applications
  – speech recognition
  – image captioning
  – summarization
  – data-to-text generation
• many different kinds of attention; see Luong et al. (2015) for other variations and comparisons
• sometimes attention seems interpretable, but it can be misleading

Luong et al. (2015): Effective Approaches to Attention-based Neural Machine Translation
Attention in Machine Translation

Bahdanau et al. (2015): Neural Machine Translation by Jointly Learning to Align and Translate
Figure 1: Example output of the attention-based summarization (ABS) system. The heatmap represents a soft alignment between the input (right) and the generated summary (top). The columns represent the distribution over the input after generating each word.

Rush et al. (2015): A Neural Attention Model for Abstractive Sentence Summarization
Natural Language Inference

Figure 1: Pictoral overview of the approach, showing the *Attend* (left), *Compare* (center) and *Aggregate* (right) steps.
Alignment between the Characters and Audio

Chan et al. (2015): Listen, Attend and Spell
Image Captioning

Figure 3. Examples of attending to the correct object (white indicates the attended regions, underlines indicated the corresponding word)

A woman is throwing a **frisbee** in a park.  
A **dog** is standing on a hardwood floor.  
A **stop** sign is on a road with a mountain in the background.

A little **girl** sitting on a bed with a teddy bear.  
A group of **people** sitting on a boat in the water.  
A giraffe standing in a forest with **trees** in the background.

*Xu et al. (2015): Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*