Sequence Model
Announcements

1. Makeup exam Dec 11

2. We will release the last reading quiz today
Recap on Convolutional neural network

Learned feature representations in CNN

[https://distill.pub/2017/feature-visualization/]
Objective today

Understanding neural network structures that are suitable for natural language (i.e., sequences of words)
Outline today

1. Word-2-Vec embedding and positional embedding

2. Attention model

3. Putting things together: the Transformer model
Example: autocompletion

e.g., I went to the climbing gym and I ___

A Language model is a conditional probability model:

\[
y_1 \sim P(Y = \cdot \ x_1, \ldots, x_n) \in \mathbb{R}^{100k}
\]

\[
y_2 \sim P(Y = \cdot \ x_1, \ldots, x_n, y_1)
\]

\[
y_m \sim P(Y = \cdot \ x_1, \ldots, x_n, y_1, \ldots y_{m-1})
\]
Word to Vector Embedding

ML models only take vectors of real numbers as inputs...

e.g., I went to the climbing gym and I

\[ u_I \in \mathbb{R}^{128} \quad u_{went} \in \mathbb{R}^{128} \quad u_{and} \in \mathbb{R}^{128} \quad u_I \in \mathbb{R}^{128} \]

Size of the English vocabulary (e.g., 100k)

128

...
Positional embedding

Order of the words and their positions matter...

e.g., When I say Transformer in ML, I do not mean the transformer in the movies

\[ u_{\text{transformer}} \in \mathbb{R}^{128} \]
\[ u_{\text{transformer}} + p_4 \in \mathbb{R}^{128} \]
\[ u_{\text{transformer}} + p_{13} \in \mathbb{R}^{128} \]

Create positional embedding using sin functions

\[
p_t = \begin{bmatrix} 
\sin(t/c_1) \\
\sin(t/c_2) \\
\vdots \\
\sin(t/c_{128}) 
\end{bmatrix}
\]

High frequency

Low frequency
Summary so far

We turn words into vectors of real numbers

e.g., When I say Transformer in ML, I do not mean the transformer in the movies

\[ u_{\text{transformer}} + p_4 \]

Feature of the word + feature of the position

\[ u_{\text{transformer}} + p_{13} \in \mathbb{R}^{128} \]
Outline today

1. Word-2-Vec embedding and positional embedding

2. Attention model

3. Putting things together: the Transformer model
Motivation

e.g., When I say Transformer in ML, I do not mean the transformer in the movies

 e.g., When I say Transformer, I literally mean the transformer in the movies

Contextual feature: feature of a word should depend on the context around it
Attention head:
three matrices:
\[ W_q, \ W_k, \ W_v \]

\[
q = W_q x \quad k = W_q x \quad v = W_q x
\]

Query  key  value

Word-2-vec + positional

Self-attention layer

I went to the climbing gym

\[
\begin{align*}
(q_1, k_1, v_1) & \quad (q_5, k_5, v_5) & \quad (q_6, k_6, v_6) \\
k_1^T q_5 & \quad \cdots & \quad k_i^T q_5 & \quad \cdots & \quad k_6^T q_5 \\
\end{align*}
\]

Softmax:
\[
p_{i,5} = \exp(k_i^T q_5) / \sum_{j=1}^{6} \exp(k_j^T q_5)
\]

\[
p_{1,5}, p_{2,5}, \ldots, p_{6,5}
\]

\[
x_5' = p_{1,5}v_1 + p_{2,5}v_2 + \ldots + p_{6,5}v_6
\]
Multi-head self-attention

I went to the climbing gym

Word-2-vec + positional

Self-attention layer \((W_q', W_k', W_v')\)

Self-attention layer \((W_q, W_k, W_v)\)
Summary so far

I went to the climbing gym

Word-2-vec + positional

Multi-head Self-attention layer \((W_q^i, W_k^i, W_v^i)_{i=1}^3\)

Contextual features: e.g., \(x_4'\) encodes information from all words
Outline today

1. Word-2-Vec embedding and positional embedding

2. Attention model

3. Putting things together: the Transformer model
The Transformer model: encoder

I went to the climbing gym

Word-2-vec + positional

Mutt-head Self-attention layer \( (W^q_i, W^k_i, W^v_i)_{i=1}^3 \)

\( x_1' + x_1 \)
\( x_2' + x_2 \)
\( x_3' + x_3 \)
\( x_4' + x_4 \)
\( x_5' + x_5 \)
\( x_6' + x_6 \)

\( x_1 \)
\( x_2 \)
\( x_3 \)
\( x_4 \)
\( x_5 \)
\( x_6 \)

Layer 1

\( \times N \) (e.g., \( N = 6 \))
The Transformer model: decoder

I went to the climbing gym

Transformer Encoder

and I trained

Word2vec + positional

\[ u_1 u_2 u_3 \]

\[ q_1 = W_q u_1 \]

\[ x_7 = \sum_{i=1}^{6} p_i v_i \]

cross-attention \((W_q, W_k, W_v)\)
The Transformer model: decoder

I went to the climbing gym

Transformer Encoder

and I trained

Word2vec + positional

Note: we do not pay attention to future words

cross-attention \( (W_q, W_k, W_v) \)
The Transformer model: decoder

I went to the climbing gym

and I trained really

Word2vec + positional

Multi-head cross-attention
\((W_q^i, W_k^i, W_v^i)_{i=1}^3\)
+residual connection and FC

Transformer Encoder

\(x_1, x_2, x_3, x_4, x_5, x_6\)

\(x_7, x_8, x_9\)

Linear classifier w/ 100k labels

\(\{p_1, p_2, \ldots, p_{100k}\} \sim \text{really}\)
Take home task:

Check out the original paper (not too hard to read!)

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez†
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin†
illia.polosukhin@gmail.com