Notes on Transformer

1 Attention Mechanism

Mimics the retrievel of a value v_i for a query q based on a key k_i in database

$$attention(q, k, v) = \sum_i similarity(q, k_i) \times v_i \tag{1}$$

1.1 Output

$$s_i = f(q, k_i) = \begin{cases} q^T k_i, \\ q^T k_i / d, \ \text{d: dimensionality of each key} \\ q^T W \cdot k_i, \ \text{general dot product: W is a weight matrix} \\ w_q^T q + w_k^T k_i, \ \text{additive similarity} \end{cases} \tag{2}$$

W maps query into the new space that could be interpreted as having the same connotation as k_i.

1.2 Softmax Layer

$$a_i = \frac{\exp s_i}{\sum_j \exp s_j} \tag{3}$$

1.3 Attention Value

$$\text{Attention} = \sum_i a_i s_i \tag{4}$$

2 Transformer

2.1 Encoder

Multi-head attention $\Longrightarrow$ Add & Norm $\Longrightarrow$ Feed & forward $\Longrightarrow$ Add & Norm

2.2 Decoder

Masked multi-head attention $\Longrightarrow$ Add & Norm $\Longrightarrow$ multi-head attention $\Longrightarrow$ Add & Norm $\Longrightarrow$ Feed & forward $\Longrightarrow$ Add & Norm $\Longrightarrow$ Linear $\Longrightarrow$ Softmax

2.3 Multihead Attention

compute multiple attentions per query with different weights

$$\text{multihead}(Q, K, V) = W^0 \text{concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h) \tag{5}$$

in which $\text{head}_i = \text{attention}(W_i^Q Q, W_i^K K, W_i^V V)$ and $\text{attention}(Q, K, V) = \text{softmax}(\frac{Q^T K}{\sqrt{d_k}})V$. Multiple heads are alike multiple filters in CNN.

2.4 Masked Multihead Attention

Some values are masked(probabilities of masked values are nullified to prevent them from being selected). When decoding, an output value should only depend on previous outputs(not future outputs), hence we mask future outputs.

$$\text{maskedAttention}(Q, K, V) = \text{softmax}(\frac{Q^T K + M}{\sqrt{d_k}}) \tag{6}$$

where M is a mask matrix of 0's and $-\infty$'s. (alike dropout, but here we try to remove connections that have not been produced yet in the sequence)

2.5 Teacher Forcing

Use correct output for decoder input (removes sequential dependence)

2.6 Other Layers

2.6.1 Layer Normalization

- Normalize values in each layer to have 0 mean and 1 variance
- For each hidden unit h_i compute $h_i \longleftarrow \frac{g}{\sigma}(h_i - \mu)$,
  where g is a variable, $\mu = \frac{1}{H}\sum_{i=1}^{H} h_i$ and $\sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(h_i - \mu)^2}$
- this reduces covariate shift (gradient dependencies between each layer and therefore fewer training iterations are needed.

2.6.2 Positional embedding

After input embedding

$$PE_{position,2i} = \sin(\frac{position}{10000^{2i/d}}) \tag{7}$$

$$PE_{position,2i+1} = \cos(\frac{position}{10000^{2i/d}}) \tag{8}$$

2.7 Complexities

Layer type	Complexity per layer	Sequential operations	Maximum path length
Self attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$
Self attention(restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

in which n is the sequence length, d is the representation dimension, and k is the kernel size of the convolution, and r is the neighborhood in restricted self-attention.

2.8 GPT & GPT-2

Decoder transformer that predicts next word based on previous inputs

2.9 BERT

Bidirectional Encoder Representation from Transformers

- Decoder transformer that predicts a missing word based on surrounding words
- mask missing word with masked multi-head attention
- improved state of the art on 11 tasks

References

[1] Vaswani et al., (2017) Attention is all you need

[2] Radford et al., (2018) Language models are unsupervised multitask learners

[3] Devlin et al., (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding