Week 1 - NMT - Encoder/Decoder with Attention

NMT : Neural Machine Translation; use of Neural Networks for Translation

Seq2Seq model was introduced by Google in 2014.

Features

- Maps a variable-length sequence to a fixed-length sequence
- Input and output don't need to have same lengths!

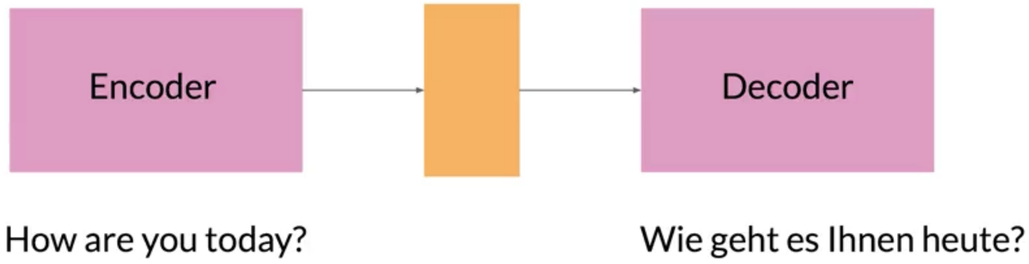

Figure 1

Problem: Later inputs in the sequence are given more importance.

Potential Solution: Instead of one vector as the context vector, pass states from individual vectors.

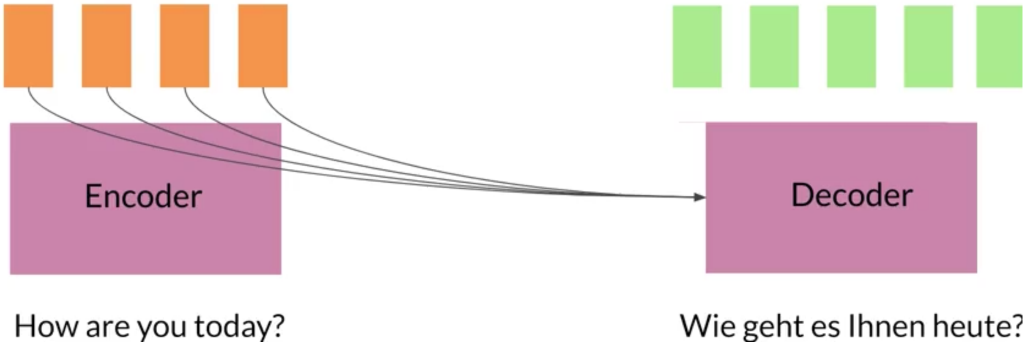

Figure 2

But this has memory constraints!

Solution: Give the model a hint to focus at the likeliest word at each step.

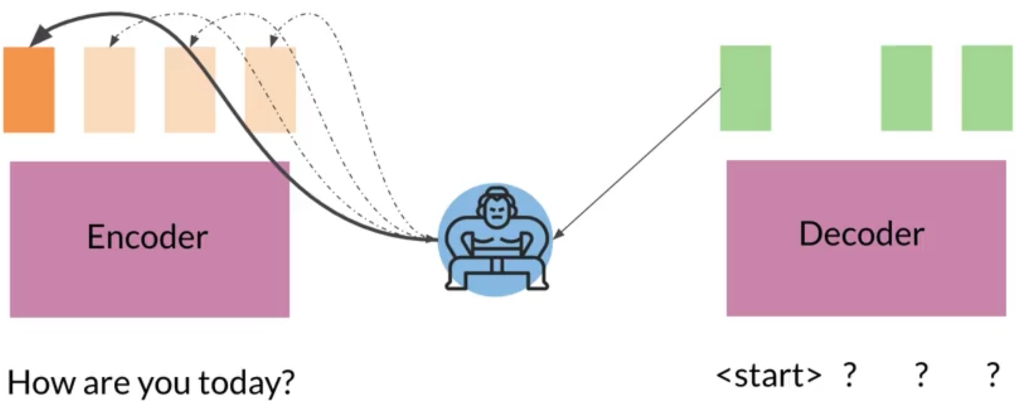

Figure 3

How to calculate using the alignment layer?

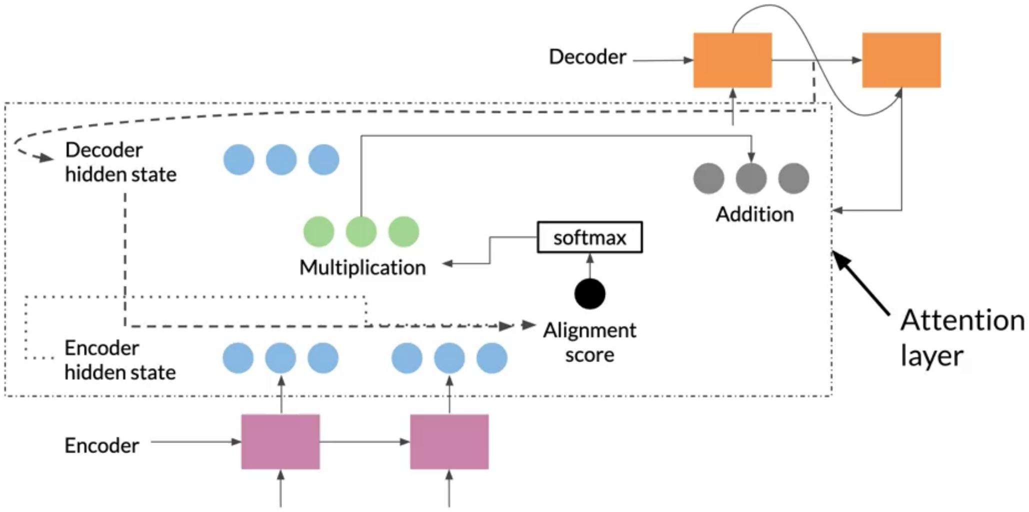

Figure 4

The complete picture

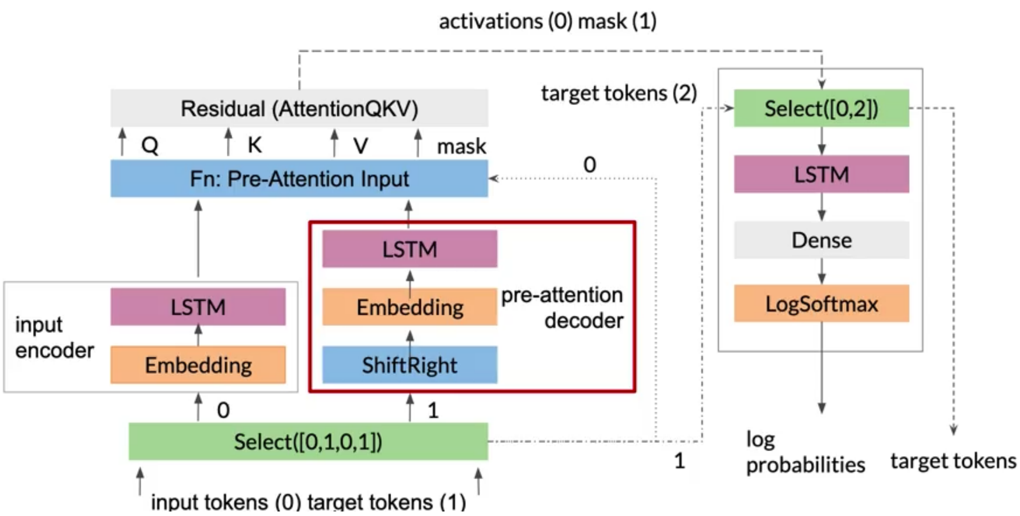

Figure 5

BLEU Scores

BLEU : Bilingual Evaluation Understudy

It evaluates the quality of machine translation by comparing it with (one or more) 'reference' translations.

Candidate	I I am I I
Reference 1	Younes said I am hungry
Reference 2	He said I am hungry

It computes two parameters

(1) brevity penalty (2) clipped precision

clipped precision

Consider unigram sequences ...

➤ Initialize a counter for the unigrams (for both predicted and actual translation). Note that the counter has unique keys only. All values are counts.

- For each word in the prediction, check if it is in the actual translation.
  - If it is absent, set the value to 0 (in the prediction key)
  - If it is present, ensure that the actual translation's count is less than or equal to the prediction count. 'Clip' if needed.

- Sum all the unigram scores of the prediction and divide by the total number of words in the prediction.
- Repeat this for bigrams, trigrams, and 4-grams.
- Weight all the scores equally with their logarithms.
- Sum the result and then take the exponent

brevity penalty

It states that the prediction sentence must be longer than the actual sentence's length. If not, scale the clipped precision by a factor

$$BP = \exp\left(1 - \frac{act\_length}{pred\_length}\right)$$

Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

Table 1: Taken from Google

ROUGE

ROUGE : Recall-Oriented understudy for Gisting Evaluation

It has two parameters

- Recall: How much of the actual text is the prediction capturing?
- How much of the prediction is relevant?

Recall:

$$R = \frac{prediction \cap actual}{no.of words in actual}$$

(Note that if our prediction was non-sensically very large, it will overlap the words in prediction and have recall = 1)

Precision:

$$P = \frac{prediction \cap actual}{no.of words in prediction}$$

The rouge score is given by

$$R = 2 * \frac{P * R}{P + R}$$

Choosing Outputs

- Greedy Decoding
- Beam-Search Decoding
- Minimum Bayes Risk

Greedy Decoding

At each step, chose the most probable word.

Beam-Search Decoding

At each step, chose the k most probable words and keep building a tree of possible words!

Minimum Bayes Risk

Generate several random sample, and assign (ROUGE) scores to each of them. Choose the sample with the highest score!

Week 2 - Transformers

Issues with RNNs

- No parallelization
- loss of information (due to vanishing gradients)

Transformers

- Don't use recurrence relation
- Use self-attention

Since there is no recurrence, there is no notion of timing
$\rightarrow$ use positional embedding to enforce timing information into the input sequence

Three ways of attention

Encoder/Decoder Attention: The Query and Key/Value belong to different sequences (target and input respectively)

Causal/Self-Attention: Query/Key/Value belong to the same sequence, but in the output block, they only look back in time

Bi-directional Self-Attention: Self-Attention without causality

The concept of multi-head is similar to that of multiple kernels in CNN.

Dimensions:

Layer/Data	Dimensions
Input(Q,K,V)	[batch_size, length, d_model]
Linear	[batch_size, length, n_heads * d_heads]
Split & Transpose	[batch_size, n_heads, length, d_heads]
Attention Result	[batch_size, n_heads, length, d_heads]
Transpose & Concat	[batch_size, length, n_heads * d_heads]
Linear	[batch_size, length, d_model]

Table 2: Dimensions

Note that

- d_model is the embedding dimension and is 512, 1024, ...
- n_heads is the number of parallel (independent heads) and is 4, 6, 16 ...
- d_heads is the dimension of each head and is 64, 128, ...

Transpose is needed to ensure that different heads (just like different batches) don't interact with each other.

Week 3 - T5/BERT

BERT has bidirectional attention.

Why Transfer Learning?

- Reduce training time
- Improve predictions
- Small datasets

While fine-tuning a pre-trained model, we add a final layer (which may have different number of outputs) and train it only.

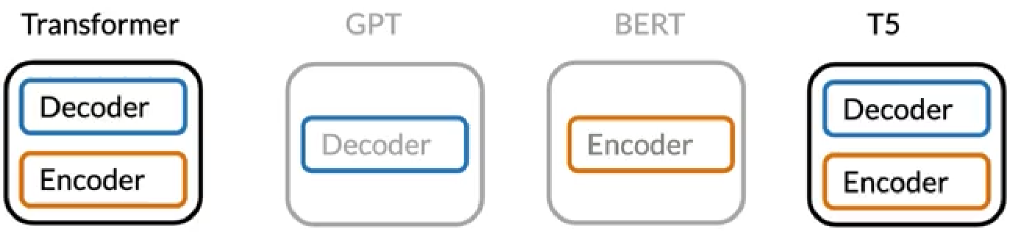

Figure 6

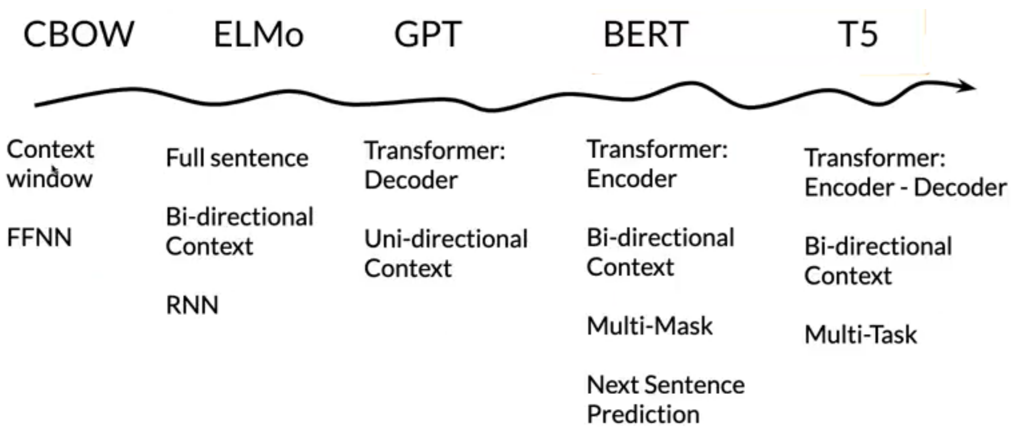

Figure 7

Week 4 -

Real world problems involve longer sequences. Addressing them requires a number of layers in Transformers.

Transformer Issues

- ➤ Attention on sequences with length L requires L^2 time and memory
- ➤ N layers takes N times as much memory

Need to find a way to minimize the compute and memory cost.

To solve this, we use reversible layers.

The Reformer is a transformer that uses reversible layers.