Exercise 6

Arnas Šniokaitis

January 5, 2026

1 2.1 Seq2seq, Attention, and Transformers

- Briefly discuss the core difference of sequence-to-sequence models compared to "normal" RNNs.

  Sequence to sequence models separate the processing of the input to create a hidden state and the usage of the hidden state to generate outputs.

- A colleague of yours proposes to use to use an RNN-based Seq-to-Seq model for German to English translation. 1) Describe what the difference to an attention-based approach is and 2) briefly discuss why you would suggest use an attention-based approach.

  Non attention based approaches can only use the hidden layer of the last input proccessing step as such it is harder to learn dependencies that are far away, where as attention based models allow the model to use all of the hidden states that have been created after each "proccessing" of the input. An attention based architecture would be prefered because, for instance; the prefixes of verbs in German separate when they are used in the indicative mood and are placed in the end of the clause. If the sentence is long enough sequence to sequence models will have a harder time learning the dependencies of verbs for correct translation where as an attention based model would be able to learn it more easily.

- Explain the difference between the cross-attention and self-attention layer in the trans- former architecture.

  Cross attention is the attention of two different sequences. Here those sequences are the input and output sequences. Self attention is attention of a sequence to itself.

  In the example given above where the goal of the model is to translate from German to English in the sentence "Wir gehen da." Self attention will learn that the verb 'gehen' is related to 'wir' for example because it denotes the ending of the verb. Where as cross attention would learn to relate the verb 'gehen' to the words in the output sequence(translation) 'are', 'going'.

- 1) Summarize the core idea of applying transformers to vision tasks (for example image classification). 2) Explain why we typically need more data to train vision transformers compared to CNN-based architectures.

The main benefit is that we can get rid of some induced biases by interpreting images as a ‘stream‘ of smaller sized image patches. This allows the model to not have as many inductive biases and thus has a bigger capacity.

Visual transformers typically need more data because they need to learn relationships that standard CNN models have inheritely due to the inductive biases that they have(for instance locality).

2 2.2 Practicing byte-pair encoding

- Describe the concept of character-level and word-level tokenization.

  We want to transform text to a sequence. To do this we need to decide how to split it into one. The idea behind character or word level tokenization is that we have a vocabulary of characters(accordingly words) which we use as tokens and split the text into these tokens.

- Discuss why these strategies are not optimal and why we typically instead use advanced tokenization approaches like byte-pair encoding.

  This is because they are not universal enough. Character level tokenization can break with accents or foreign words. For example in british english naive is more commonly spelt as naïve to show that the word is borrowed from French. If we use the standard English alphabet as our tokens it will not be able to translate this word into a sequence. Character level tokenization breaks in a simmilar way. Word level tokenization also fails when new words are created as they will not be in the vocabulary of the model and the model will need to be retrained. Another issue is memory. For character level tokenization medium sized sentecnes can turn into massive sequences and word tokenization will have a massive vocabulary both of which are not ideal.

Step	Tokens	Tokenized Text	Frequencies	Merge Rule
#1	S, L, E, P, S, N, (w)	'S', 'L', 'E', 'E', 'P', 'L', 'E', 'S', 'S', 'N', 'E', 'S', 'S', '(w)'	'SL'-1, 'LE'-2, 'EE'-1, 'EP'-1, 'PL'-1, 'ES'-2, 'SS'-2, 'SN'-1, 'NE'-1, 'S(W)'-1	'E' + 'S' → 'ES'
#2	S, L, E, P, S, N, (w), ES	'S', 'L', 'E', 'E', 'P', 'L', 'ES', 'S', 'N', 'ES', 'S', '(w)'	'SL'-1, 'LE'-1, 'EE'-1, 'EP'-1, 'PL'-1, 'LES'-1, 'ESS'-2, 'SN'-1, 'NES'-1, 'S(W)'-1	'ES' + 'S' → 'ESS'
#2	S, L, E, P, S, N, (w), ES, ESS	'S', 'L', 'E', 'E', 'P', 'L', 'ESS', 'N', 'ESS', '(w)'	'SL'-1, 'LE'-1, 'EE'-1, 'EP'-1, 'PL'-1, 'LESS'-1, 'ESSN'-1, 'NESS'-1, 'ESS(w)'-1	'E' + 'E' → 'EE'

3 Training of the model

Below is the predictions that the model provided after training for 10 epochs.

```
Sample 1:
Input:     [13, 43, 44, 95, 99, 36, 41, 47, 50, 49, 71, 21, 18, 70, 43, 24, 26, 68, 34, 65]
Target:    [43, 44, 95, 99, 36, 41, 47, 50, 49, 71, 21, 18, 70, 43, 24, 26, 68, 34, 65, 0]
Predicted: [43, 44, 95, 99, 36, 41, 47, 50, 49, 71, 21, 18, 70, 43, 24, 26, 68, 34, 65, 0]
----------------------------------------
Sample 2:
Input:     [51, 40, 12, 12, 34, 80, 74, 59, 35, 49, 22, 52, 68, 52, 19, 70, 16, 71, 93, 58]
Target:    [40, 12, 12, 34, 80, 74, 59, 35, 49, 22, 52, 68, 52, 19, 70, 16, 71, 93, 58, 0]
Predicted: [40, 12, 12, 34, 80, 74, 59, 35, 49, 22, 52, 68, 52, 19, 70, 16, 71, 93, 58, 0]
----------------------------------------
Sample 3:
Input:     [88, 75, 45, 68, 92, 88, 38, 40, 86, 15, 97, 46, 8, 35, 14, 71, 80, 4, 59, 58]
Target:    [75, 45, 68, 92, 88, 38, 40, 86, 15, 97, 46, 8, 35, 14, 71, 80, 4, 59, 58, 0]
Predicted: [75, 45, 68, 92, 88, 38, 40, 86, 15, 97, 46, 8, 35, 14, 71, 80, 4, 59, 58, 0]
----------------------------------------
Sample 4:
Input:     [58, 4, 6, 21, 74, 3, 60, 10, 17, 64, 83, 31, 95, 37, 22, 46, 15, 4, 59, 95]
Target:    [4, 6, 21, 74, 3, 60, 10, 17, 64, 83, 31, 95, 37, 22, 46, 15, 4, 59, 95, 0]
Predicted: [4, 6, 21, 74, 3, 60, 10, 17, 64, 83, 31, 95, 37, 22, 46, 15, 4, 59, 95, 0]
----------------------------------------
Sample 5:
Input:     [37, 46, 70, 86, 30, 36, 19, 69, 83, 38, 32, 25, 8, 86, 18, 68, 33, 21, 33, 69]
Target:    [46, 70, 86, 30, 36, 19, 69, 83, 38, 32, 25, 8, 86, 18, 68, 33, 21, 33, 69, 0]
Predicted: [46, 70, 86, 30, 36, 19, 69, 83, 38, 32, 25, 8, 86, 18, 68, 33, 21, 33, 69, 0]
----------------------------------------
```

Looking at the graph of the error/loss we can see that by the 10-th epoch the error is relatively small which explains the good predictions.

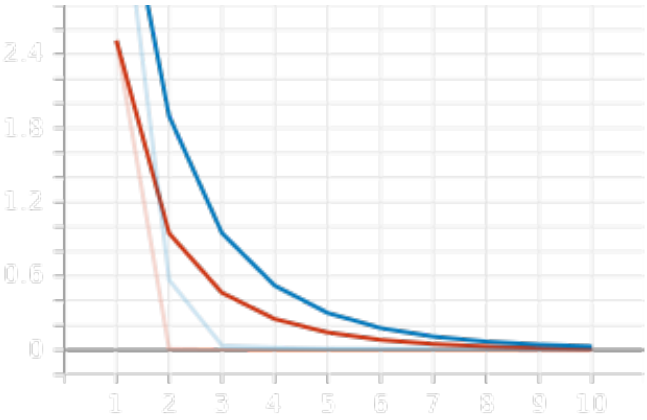

Figure 1: Training and validation losses

Repeated training of the model does not yeald drastically different results as they follow the same rough curve and become roughly equal with the losses being close to 0 by the 10-th epoch as such they will not be included.