Assignment 4 Writeup

DO NOT TAG

Name:
GT Email:

Seq2Seq Results – Default configuration

RNN

Training Loss: 4.2824
Training Perplexity: 72.4149
Validation Loss: 4.3507
Validation Perplexity: 77.5293

LSTM

Training Loss: 3.0209
Training Perplexity: 20.5091
Validation Loss: 3.1772
Validation Perplexity: 23.9794

RNN-with-Attention

Training Loss: 3.1348
Training Perplexity: 22.9838
Validation Loss: 3.2645
Validation Perplexity: 26.1675

LSTM-with-Attention

Training Loss: 3.0325
Training Perplexity: 20.7501
Validation Loss: 3.1941
Validation Perplexity: 24.3877

Seq2Seq Explanation (RNN vs LSTM)

Compare your RNN result to your LSTM result and explain why they differ.

The LSTM model achieved significantly lower training and validation losses and perplexities compared to the RNN.

LSTMs are designed to handle long-term dependencies more effectively than standard RNNs. The primary reason for this difference lies in the way LSTMs manage the flow of information over time using gating mechanisms (input, forget, and output gates). These gates allow LSTMs to better control which information is remembered or forgotten, reducing the impact of vanishing gradients and allowing the network to capture more complex patterns in sequential data. In contrast, vanilla RNNs lack these gating mechanisms, making them more prone to issues with long sequences, particularly vanishing or exploding gradients, which can hinder learning. This is especially problematic in language models where dependencies between words can span long sequences. As a result, the LSTM can learn and retain information across longer time steps, leading to more accurate predictions and, therefore, lower loss and perplexity compared to the RNN. This makes LSTMs generally more effective for tasks involving sequences with complex dependencies, such as language modeling.

Our LSTM outperformed our RNN across the board

Seq2Seq Explanation (RNN vs RNN-with-Attention)

Compare your RNN result to your RNN-with-Attention result and explain why they differ.

The comparison between our RNN and RNN-with-Attention results shows that adding attention improves performance, as evidenced by the lower training and validation losses and perplexities for the RNN with attention

The attention mechanism allows the model to focus selectively on specific parts of the input sequence when generating each output word. This selective focus helps the model capture relevant information from distant parts of the sequence more effectively, which is especially valuable in cases where long-term dependencies are important, such as language modeling. Without attention, a standard RNN relies entirely on the hidden state at each time step, which can lead to issues with capturing long-range dependencies due to the vanishing gradient problem. Attention mitigates this issue by providing a weighted combination of all hidden states across the sequence, making it easier to maintain relevant context over longer sequences. As a result, the RNN-with-Attention can learn more efficiently and make more accurate predictions, leading to lower perplexities and losses compared to the standard RNN. This improvement demonstrates the value of attention in enhancing the expressiveness and effectiveness of recurrent models.

Seq2Seq Results – Best model

Your best model after hyper-parameter tuning

Best model

Training Loss: 2.3803
Training Perplexity: 10.8082
Validation Loss: 3.0843
Validation Perplexity: 21.8530

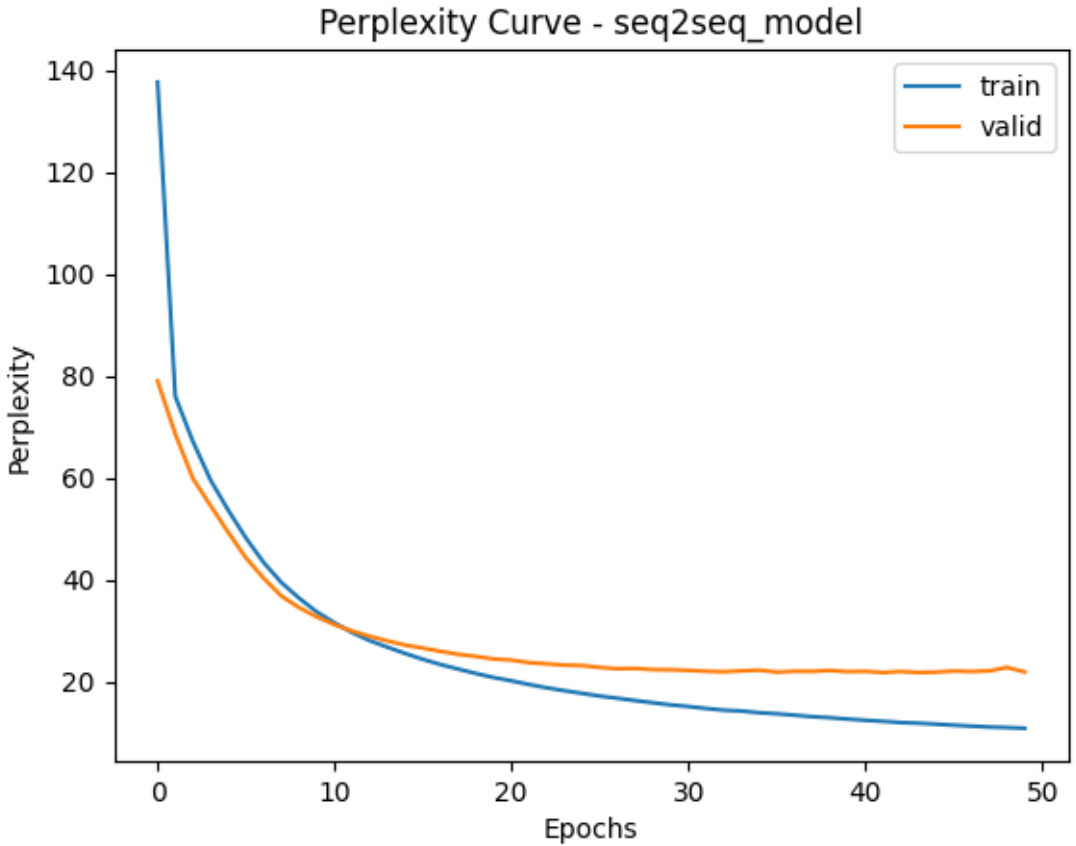
List your best model hyper-parameter values including model type:

encoder_emb_size = 256
encoder_hidden_size = 256
encoder_dropout = 0.15

decoder_emb_size = 256
decoder_hidden_size = 256
decoder_dropout = 0.15

learning_rate = 1e-2
model_type = "LSTM“
Attention = True
Epochs = 50

Seq2Seq Best model Learning Curves (Perplexity)



Seq2Seq Explanation – Best model

Explain the details of your best model. Explain what you did to improve your model's performance and why

My best model configuration was as follows:
Encoder and Decoder Embedding Sizes: 256
Encoder and Decoder Hidden Sizes: 256
Dropout: 0.15
Learning Rate: 0.01
Model Type: LSTM (with attention)
Epochs: 50

To improve my model's performance, I tuned various hyperparameters, starting with the embedding and hidden sizes, which I increased from the default values (128) to 256. This adjustment aimed to provide the model with a richer representation, potentially helping it capture more complex patterns. I also reduced the dropout from 0.2 to 0.15, based on experimentation to find an optimal balance between regularization and network capacity. I also lowered the learning rate. Overall, this sped up learning though this had minimal improvement at longer epochs. We also found that the LSTM with attention model gave us the best results.

However, training was unstable, and improving upon the default settings proved challenging. By the end of training, the model displayed significant overfitting, evident in the widening gap between training and validation metrics. While I experimented with higher dropout rates to reduce overfitting, performance gains were limited.

Overall though, we were still able to improve the validation accuracy and perplexity materially. The increased encoder decoder sizes gave the model more power to make accurate predictions.

Transformer Results

Default configuration (Encoder Only)

Training Loss: 2.1373
Training Perplexity: 8.4769
Validation Loss: 2.9339
Validation Perplexity: 18.8005

Default configuration (full transformer)

Training Loss: 1.3301
Training Perplexity: 3.7815
Validation Loss: 1.5861
Validation Perplexity: 4.8847

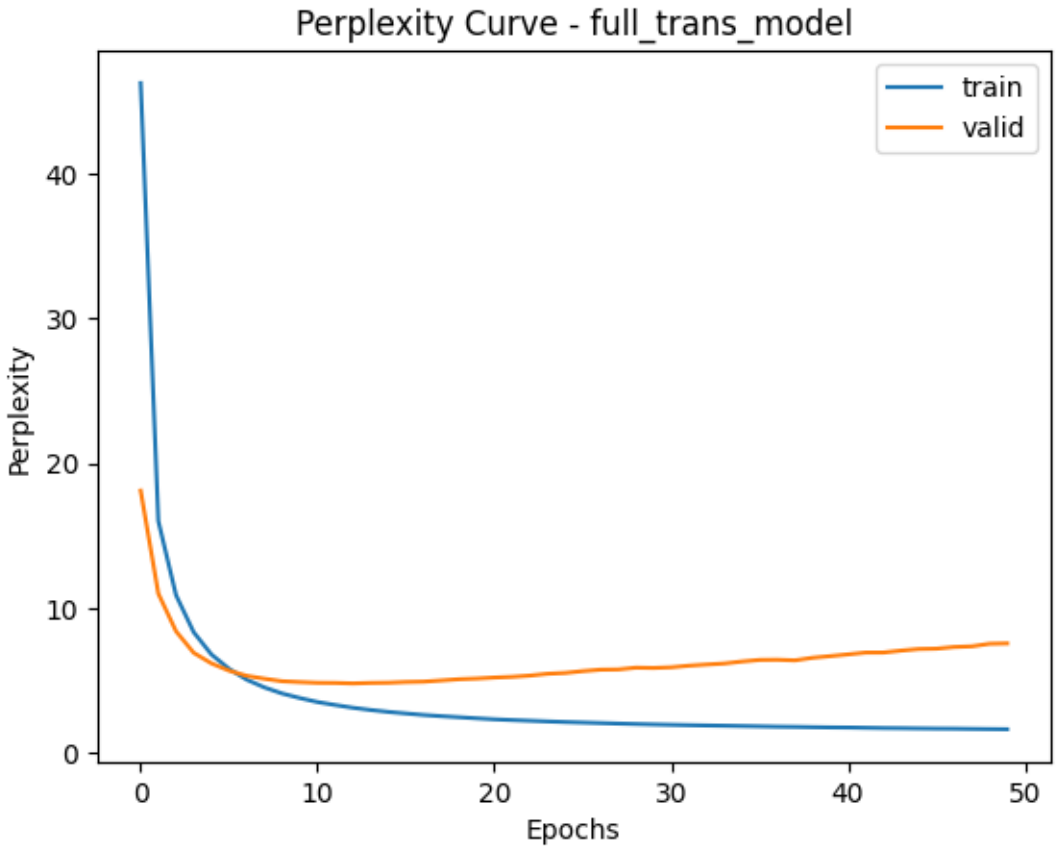
Best model (full transformer)

Training Loss: 1.1349.
Validation Loss: 1.5669.
Training Perplexity: 3.1109.
Validation Perplexity: 4.7920
(epoch 14)

List your best model hyper-parameter values (full transformer):

learning_rate = 0.0005
Epochs = 50
(No change to neural net dimensions)

Full Transformer Best model Learning Curves (Perplexity)



Full Transformer Explanation – Best model

Explain what you did here and why you did it to improve your model performance.

To improve the performance of our transformer model, we only ended up moving the learning rate slightly lower to 0.0005.

The default parameters were remarkably robust to any changes we made, but we were able to improve them slightly. By lowering the learning rate, I aimed to make the model's updates more stable and precise, which is often beneficial for transformers where high learning rates can lead to erratic training and suboptimal convergence. Reducing the learning rate helped the model to better capture patterns in the data by taking smaller, more controlled steps towards optimization. I also adjusted the encoder and decoder dimensions to explore if different model capacities could enhance learning. Increasing or decreasing the dimensions can influence the model's capacity to represent complex data patterns; however, I found that extensive changes led to limited improvement, possibly due to the increased risk of overfitting or capacity misalignment with the dataset.

Even the improvements we found eventually succumb to overfitting as the validation performance started to increase away from the training performance. By incorporating early stopping, we find our ideal model for generalization at the 14^{th} epoch which is where the scores come from.

Overall, while these adjustments were intended to improve the model, I still faced challenges in achieving substantial improvements beyond the default parameters, suggesting that the model may be limited by other factors in the architecture or the data itself or the default parameters were particularly suitable to this experiment.

Transformer (Encoder Only) Translation Results

Put translation results for your model (1st 9 sentences) here

True Translation	Predicted Translation
‘<sos>'word_1’,’word_2’,….., '<eos>'	‘<sos>'word_1’,’word_2’,….., '<eos>'
'<sos>', 'a', 'man', 'in', 'an', 'orange', 'hat', 'starring', 'at', 'something', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>']	'<sos>', 'a', 'man', 'in', 'an', 'orange', 'hat', 'an', 'at', '.', '.', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'boston', 'terrier', 'is', 'running', 'on', 'lush', 'green', 'grass', 'in', 'front', 'of', 'a', 'white', 'fence', '.', '\n', '<eos>', '<pad>'	'<sos>', 'a', 'soccer', ',', 'about', 'front', 'across', 'grass', 'fence', 'of', 'front', 'white', 'a', 'fence', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'girl', 'in', 'karate', 'uniform', 'breaking', 'a', 'stick', 'with', 'a', 'front', 'kick', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'girl', 'girl', 'a', 'goggles', 'a', 'to', 'with', 'with', 'a', 'nike', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'people', 'are', 'fixing', 'the', 'roof', 'of', 'a', 'house', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'people', 'repair', 'the', 'the', 'a', 'a', 'a', 'shoveling', '.', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'group', 'of', 'people', 'standing', 'in', 'front', 'of', 'an', 'igloo', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'group', 'of', 'people', 'standing', 'in', 'front', 'front', 'a', 'a', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'guy', 'works', 'on', 'a', 'building', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'guy', 'is', 'working', 'a', 'a', 'building', 'building', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'in', 'a', 'vest', 'is', 'sitting', 'in', 'a', 'chair', 'and', 'holding', 'magazines', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'in', 'a', 'vest', 'vest', 'is', 'sitting', 'a', 'holding', 'and', 'and', '\n', '\n', '\n', '\n', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'mother', 'and', 'her', 'young', 'song', 'enjoying', 'a', 'beautiful', 'day', 'outside', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'mother', 'and', 'her', 'mother', 'enjoying', 'enjoying', 'day', 'a', 'sunny', 'day', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
‘<sos>', 'a', 'woman', 'holding', 'a', 'bowl', 'of', 'food', 'in', 'a', 'kitchen', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'woman', 'in', 'a', 'her', 'food', 'a', 'ingredients', '\n', '.', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '\n', '<eos>'

Full Transformer Translation Results

Put translation results for your best model (1st 9 sentences) here

True Translation	Predicted Translation
‘<sos>'word_1’,’word_2’,….., '<eos>'	‘<sos>'word_1’,’word_2’,….., '<eos>'
'<sos>', 'a', 'man', 'in', 'an', 'orange', 'hat', 'starring', 'at', 'something', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>']	'<sos>', 'a', 'man', 'in', 'an', 'orange', 'hat', 'with', 'an', 'orange', 'hat', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>'
'<sos>', 'a', 'boston', 'terrier', 'is', 'running', 'on', 'lush', 'green', 'grass', 'in', 'front', 'of', 'a', 'white', 'fence', '.', '\n', '<eos>', '<pad>'	'<sos>', 'a', 'boston', 'team', 'is', 'running', 'across', 'the', 'grass', 'in', 'front', 'of', 'a', 'white', 'fence', '.', '\n', '<eos>', '\n', '<eos>'
'<sos>', 'a', 'girl', 'in', 'karate', 'uniform', 'breaking', 'a', 'stick', 'with', 'a', 'front', 'kick', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'girl', 'in', 'a', 'karate', 'uniform', 'is', 'making', 'a', 'kick', 'of', 'a', 'kick', '.', '\n', '<eos>', '.', '\n', '<eos>'
'<sos>', 'people', 'are', 'fixing', 'the', 'roof', 'of', 'a', 'house', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'people', 'repair', 'the', 'roof', 'of', 'a', 'house', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>'
'<sos>', 'a', 'group', 'of', 'people', 'standing', 'in', 'front', 'of', 'an', 'igloo', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'group', 'of', 'people', 'are', 'standing', 'in', 'front', 'of', 'a', 'motorized', 'hose', '.', '\n', '<eos>', '.', '\n', '<eos>', '.'
'<sos>', 'a', 'guy', 'works', 'on', 'a', 'building', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'guy', 'working', 'on', 'a', 'building', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>', '.'
'<sos>', 'a', 'man', 'in', 'a', 'vest', 'is', 'sitting', 'in', 'a', 'chair', 'and', 'holding', 'magazines', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'in', 'a', 'vest', 'is', 'sitting', 'on', 'a', 'chair', 'holding', '<unk>', '.', '\n', '<eos>', '.', '\n', '<eos>', '.'
'<sos>', 'a', 'mother', 'and', 'her', 'young', 'song', 'enjoying', 'a', 'beautiful', 'day', 'outside', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'mother', 'and', 'her', 'son', 'enjoying', 'a', 'beautiful', 'day', 'outside', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>'
‘<sos>', 'a', 'woman', 'holding', 'a', 'bowl', 'of', 'food', 'in', 'a', 'kitchen', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'woman', 'holding', 'a', 'kitchen', 'with', 'food', 'in', 'a', 'kitchen', '.', '\n', '<eos>', '.', '\n', '<eos>', '.', '\n', '<eos>'

Seq2Seq (Best model) Translation Results

Put translation results for your best model (1st 9 sentences) here

True Translation	Predicted Translation
‘<sos>'word_1’,’word_2’,….., '<eos>'	‘<sos>'word_1’,’word_2’,….., '<eos>'
'<sos>', 'a', 'man', 'in', 'an', 'orange', 'hat', 'starring', 'at', 'something', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>']	'<sos>', 'a', 'man', 'in', 'a', 'a', 'shirt', 'and', 'is', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'boston', 'terrier', 'is', 'running', 'on', 'lush', 'green', 'grass', 'in', 'front', 'of', 'a', 'white', 'fence', '.', '\n', '<eos>', '<pad>'	'<sos>', 'a', 'is', 'dog', 'running', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'girl', 'in', 'karate', 'uniform', 'breaking', 'a', 'stick', 'with', 'a', 'front', 'kick', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'girl', 'in', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'people', 'are', 'fixing', 'the', 'roof', 'of', 'a', 'house', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'people', 'are', 'on', 'on', 'on', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'group', 'of', 'people', 'standing', 'in', 'front', 'of', 'an', 'igloo', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'of', 'people', 'stands', 'in', 'in', 'a', 'a', '.', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'guy', 'works', 'on', 'a', 'building', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'is', 'a', 'a', 'a', '.', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'man', 'in', 'a', 'vest', 'is', 'sitting', 'in', 'a', 'chair', 'and', 'holding', 'magazines', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'man', 'in', 'a', 'a', 'a', 'sitting', 'is', 'is', 'on', 'a', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
'<sos>', 'a', 'mother', 'and', 'her', 'young', 'song', 'enjoying', 'a', 'beautiful', 'day', 'outside', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'child', 'is', 'a', 'the', 'the', 'a', 'a', 'a', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>'
‘<sos>', 'a', 'woman', 'holding', 'a', 'bowl', 'of', 'food', 'in', 'a', 'kitchen', '.', '\n', '<eos>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>'	'<sos>', 'a', 'woman', 'is', 'a', 'a', 'a', 'a', 'a', 'a', 'a', '.', '\n', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>', '<eos>']

Compare Transformer (Encoder Only) to Transformer (Full transformer)

Compare your results for default settings for Encoder Only Transformer vs Full transformer both quantitatively and qualitatively. Explain why you see differences.

Well first to state the obvious, the full transformer far outperformed the Encoder only Transformer. It performed better both quantitatively (validation loss and perplexity) and qualitatively (e.g., output coherence and relevance).

The key reason for the performance gap is that the encoder-only transformer lacks a mechanism for autoregressive output generation. While encoder-only models are efficient for tasks like classification, they generally fall short for tasks requiring sequential output, where each output token depends on both the input and preceding outputs. The full transformer's encoder-decoder setup better models these dependencies, making it more suitable for tasks with complex input-output relationships, where the quality of the output sequence (coherence and accuracy) is paramount. The full transformer's superior results stem from its architecture's ability to handle complex dependencies across both input and output sequences, which an encoder-only model cannot effectively capture.

This is easily displayed in some of the outputs from the translations. The encoder-only transformer tends to repeat words and while the content may be directionally correct, the grammatical structure is almost always jumbled as opposed to the Full-Transformer. It also seems to struggle with uncommon words like "magazines" or "lush" for instance it incorrectly translates "karate uniform" to "goggles" whereas the Full-Transformer is able to reproduce the correct word even if its translation isn't word for word.

It was really fun to see the Full Transformer actually produce useful translations. Especially after I ran the Seq2Seq model and its output was so poor I feared I had not implemented it correctly until I saw the results of the other models.

Compare Seq2Seq to Transformer (Best models)

Compare your Seq2Seq best model results to your Transformer best model results both quantitatively and qualitatively and explain the differences.

Here the Full Transformers results were far superior to the Seq2seq model both quantitatively and qualitatively. The Full-Transformer has several architectural advantages that allow the transformer to better capture context, long-range dependencies, and parallelism, all of which significantly enhance performance.

Parallel Processing: Unlike Seq2Seq models, which process tokens sequentially, transformers use self-attention to process entire sequences simultaneously. This parallelism makes transformers faster to train and better at handling long-range dependencies.
Attention Mechanism: The self-attention layers in transformers allow each token to dynamically “attend” to other tokens, capturing complex relationships across sequences more effectively. Seq2Seq models, even with attention, often struggle with maintaining long-range dependencies, especially in longer sequences.
Positional Encoding: Transformers encode positional information explicitly, enabling them to manage sequence order flexibly without depending on sequential structure as RNN-based Seq2Seq models do. This improves their ability to capture both local and global dependencies within the data.
Scalability: Transformers are more scalable, allowing for deeper and wider architectures that can capture nuanced patterns across massive datasets. This adaptability makes them ideal for tasks that demand high accuracy on extensive data.

The translations speak for themselves as the Full Transformer got pretty close to the original translation while the Seq2Seq basically got 2 to 3 words correct in the whole sequence amongst a garbled group of articles like “a” and “is”. This naturally showed up in the far inferior quantitative metrics as well.

Theory question

To compute the partial derivative $\frac{\partial L_2}{\partial U}$ for a 3-length sequence RNN, let's follow the forward and backward pass using the given equations. We will use chain rule differentiation to get the required gradients and show how we derived each part.

1. First, let's understand what ∂L2/∂U means:
We want to find how L2 (loss at t=2) changes with respect to U. U is used in the computation of a_t for all time steps. However, due to the recurrent nature, h1 affects a2, so we need to consider both paths

2. Using the chain rule, we can break this into parts:
∂L2/∂U = ∂L2/∂a2 × ∂a2/∂U (direct path) + ∂L2/∂a2 × ∂a2/∂h1 × ∂h1/∂a1 × ∂a1/∂U (indirect path through h1)

3. Let's calculate each term

a. ∂L2/∂a2 = ∂L2/∂o2 × ∂o2/∂h2 × ∂h2/∂a2 = (^y2 - y2) × V × (1 - tanh²(a2))
b. ∂a2/∂U = x2 (direct path)
c. ∂a2/∂h1 = W (from the equation at = Uxt + Wht-1 + b)
d. ∂h1/∂a1 = (1 - tanh²(a1))
e. ∂a1/∂U = x1

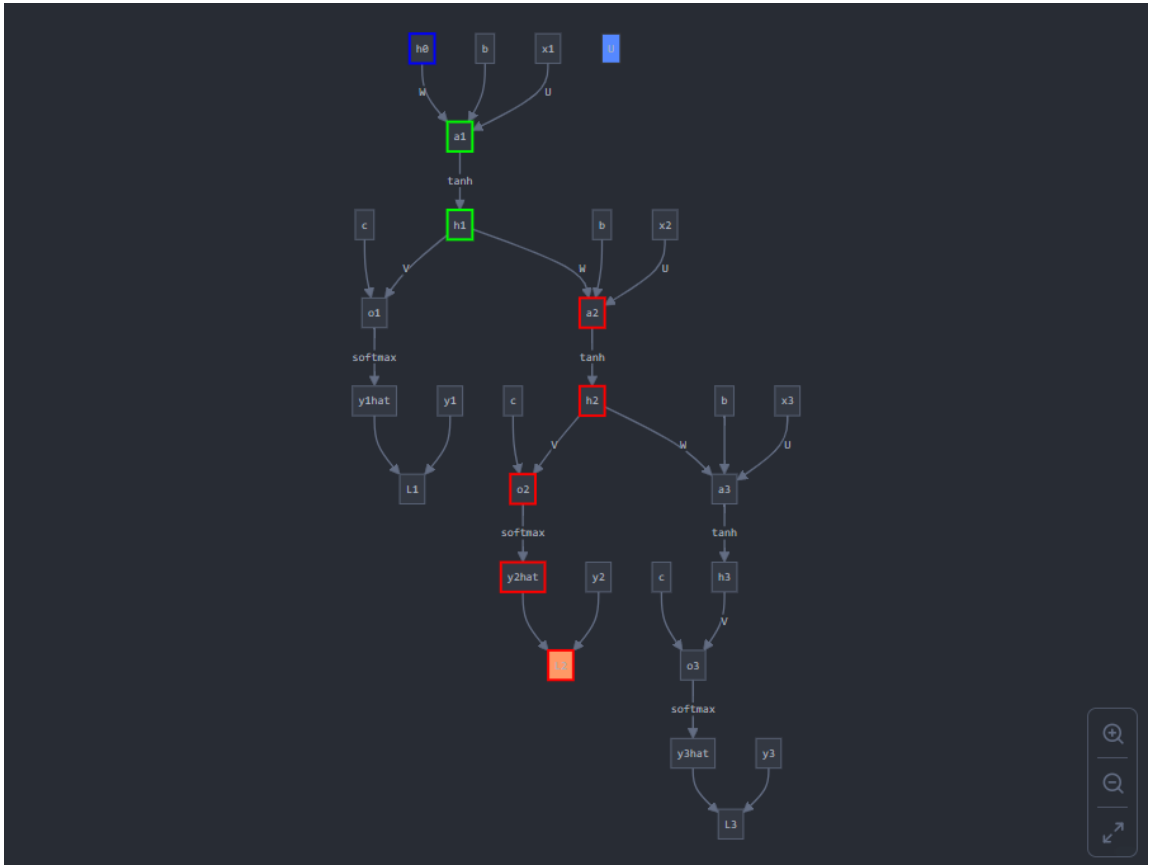
4. Putting it all together:
∂L2/∂U = (∂L2/∂o2 × ∂o2/∂h2 × ∂h2/∂a2) × x2 + (∂L2/∂o2 × ∂o2/∂h2 × ∂h2/∂a2) × W × (1 - tanh²(a1)) × x1

= (^y2 - y2)V(1 - tanh²(a2))x2 + (^y2 - y2)V(1 - tanh²(a2))W(1 - tanh²(a1))x1

5. This gives us the final gradient for ∂L2/∂U, with two terms:
First term represents direct influence of U on L2 through x2.
Second term represents indirect influence through the recurrent connection h1.
This derivation shows how the error at time step 2 propagates back through the network both directly and through the recurrent connections to affect the gradient of U.

Theory question (continued)

Computation graph:



Paper discussion

Name the paper you chose and answer the questions related to it. Add additional slides as necessary
I chose: "Do Vision Transformers See Like Convolutional Neural Networks?" by Raghu et al

Review:

The paper "Do Vision Transformers See Like Convolutional Neural Networks?" by Raghu et al. primarily investigates the representational differences between Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs) in image classification tasks. The main contribution is its analysis of how ViTs and CNNs differ in their hierarchical feature extraction and internal representations, which it quantifies using similarity measures such as Centered Kernel Alignment (CKA). The study finds that ViTs process spatial information more globally from early layers, unlike CNNs, which build local features first and gradually expand to global structures in deeper layers. This difference hints at why ViTs perform well on tasks requiring contextual understanding but may need more training data than CNNs to generalize effectively.

Strengths: The paper's strengths include its thorough quantitative approach to comparing ViTs and CNNs and its use of CKA to systematically analyze representational similarities across architectures. This approach provides a clear, data-driven insight into how ViTs' self-attention mechanism allows for non-local information processing early in the network, a strength for vision tasks involving complex, global contexts.
Weaknesses: The paper's weaknesses include the limited scope of tasks examined—primarily focused on image classification—which may not fully represent the versatility of ViTs across other vision tasks (e.g., segmentation or object detection). Additionally, while the paper highlights the global processing strengths of ViTs, it does not deeply address their limitations, such as computational cost and efficiency issues on smaller datasets.
Personal takeaway: The paper is a valuable addition to understanding architectural differences, emphasizing that ViTs' self-attention allows for a distinct, more contextually aware approach to vision tasks than the hierarchical feature-building seen in CNNs. Future research could explore hybrid models that combine ViTs' global receptive fields with CNNs' efficient, layer-by-layer processing, potentially enhancing performance in settings with less data. It was the first thought that came to me: why not use the strengths of both moving forward? Also it continues to a positive attitude towards research that we can keep looking for new and better architectures and models (ViTs) and aren't consigned to the legacy methods (CNNs) that we have found work well so far.

Paper discussion (continued)

Additional Questions:
Compare and contrast the learned features of ViTs and CNNs? For differences between the two, please provide explanations in terms of network architecture and training.

The learned features of Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs) differ significantly due to their unique architectures and methods of spatial information processing.

CNNs are built around convolutional layers that focus on extracting localized patterns and hierarchically building up from low-level to high-level features. Early layers capture edges and textures, while deeper layers assemble these into increasingly complex shapes and objects. This hierarchical structure, optimized for spatial locality, is effective for capturing fine-grained details but may struggle with long-range dependencies unless layers are stacked deeply.
In contrast, ViTs rely on self-attention, enabling them to process global information across the image from the start. In a ViT, the input image is split into patches, and the transformer layers apply attention across these patches, capturing relationships between distant regions early in the network. This global approach allows ViTs to integrate broader contextual information and can be especially powerful for tasks where understanding the spatial arrangement across an entire image is essential. However, ViTs generally require larger datasets for effective training, as they lack the built-in spatial biases of CNNs.
Architecturally, CNNs use convolution and pooling to progressively reduce spatial dimensions and expand the receptive field in a structured manner, while ViTs use multi-head self-attention without any specific inductive bias towards spatial locality. This makes ViTs more flexible but computationally heavier, as they attend to all pairs of patches in each layer. Hence the need for more training data.
In training, CNNs tend to generalize well even with smaller datasets due to their spatial bias and translation invariance. ViTs, by contrast, require extensive data or data augmentation to generalize as they learn spatial relationships from scratch. This difference reflects each model's approach: CNNs are inductively biased to detect localized patterns, whereas ViTs, with self-attention, dynamically model global relationships across the image, making them adaptable but data-intensive.

The fact that these two architectures have different strengths and weakness is a net positive as we can use an amalgamation of the two to solve different machine learning tasks based on what the solution calls for.

Paper discussion (continued)

Additional Questions:
What is meant by spatial localization? And why might we consider the use of ViTs better for object detection?

In the context of image recognition models, spatial localization refers to a model's ability to recognize and maintain the relative positions of objects or patterns within an image. For CNNs, this is achieved through the use of convolutional filters that progressively focus on localized regions, capturing patterns in a structured way, from small details in early layers to more complex spatial hierarchies in deeper layers. This spatial bias allows CNNs to identify and differentiate objects based on their relative positions, which has been foundational in tasks like image classification and detection.

Vision Transformers (ViTs), on the other hand, approach spatial localization differently. Instead of relying on localized filters, ViTs use self-attention mechanisms that can attend to all parts of the image simultaneously. This means that ViTs can capture relationships between widely separated parts of an image from the outset, enabling them to understand the overall spatial context rather than just local details. This global attention allows ViTs to consider an object as a whole rather than as a series of progressively combined features, potentially making ViTs more suited to identifying objects that might otherwise be challenging to localize based on smaller, localized filters.

For object detection, ViTs may be preferable because their self-attention mechanism enables them to understand and relate parts of an object regardless of distance or spatial separation. This makes them adept at handling complex scenes with multiple overlapping objects or objects appearing at varying scales. The ability to incorporate context from across the entire image allows ViTs to perform well in scenarios where recognizing an object requires understanding its relationship to distant elements. This could lead to more robust object detection in cases where objects might be partially obscured or spaced irregularly, providing an edge over CNNs in such tasks. ViTs are simply better equipped than CNNs to take into account the complete context of the image and thus more accurately identify and detect objects when that adjoining context is relevant.