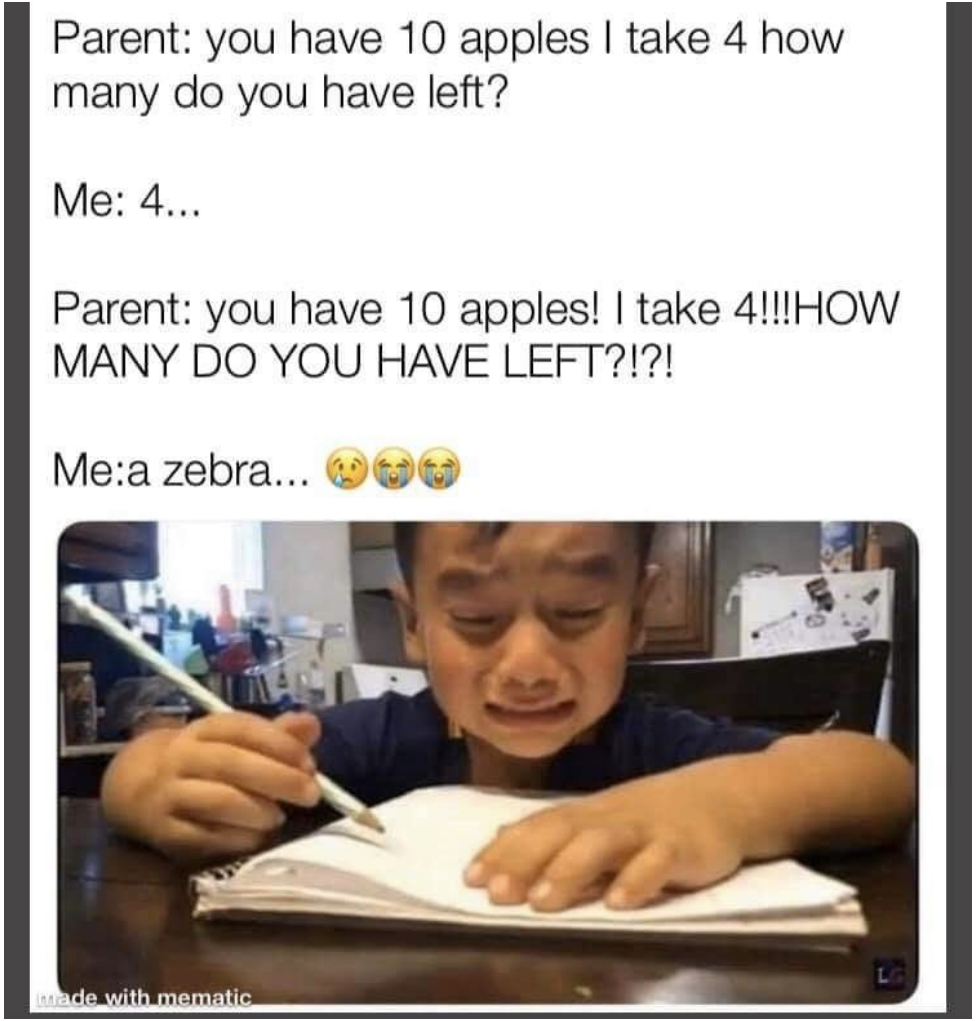
Math Word Problem to Equation

- Aashna Kanuga (adk2159)
- Akshata Patel (amp2313)

Aim:
Using DL to eradicate this issue!
:)



Source: Facebook

Introduction

Our aim with the project is to use Deep Learning to automatically convert simple math word problems, such as “If Andy has 5 apples and Jerry eats 1, how many are left?”, to a math equation: “x = 5 - 1”.

We have experimented with seq2seq models with Attention, and Transformer model to accomplish this task.

The project is broadly divided into two parts:

1. A baseline, where we use a simple list of generated problems, such as “Two hundred fifty plus Three thousand nine hundred” to measure the seq2seq model performance.
2. We then take a dataset of ~38k single variable math word problems and train a seq2seq and Transformer model and compare their performance.

Metrics used

We used two metrics to evaluate the performance of all our models:

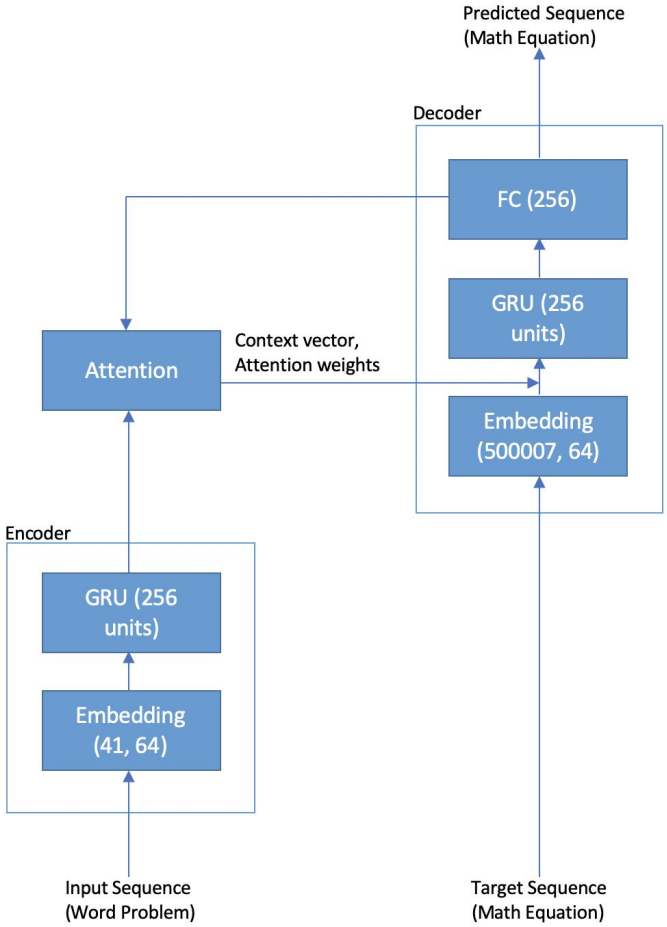
1. Accuracy: In order to see how exact the predictions of the model are
2. Corpus BLEU (Bilingual Evaluation Understudy) score: A metric developed specifically to for auto-translation systems, BLEU score compares n-grams of the candidate and reference translation, so even if the translation is not an exact match, the score is not zero.

Baseline Dataset

Baseline Model Architecture

The Seq2Seq model used has an encoder, decoder and an attention model.

- The encoder model takes in the word problem as a sequence.
- The Bahdanau Attention computes attention weights for the input sequence along with the most recent prediction.
- The decoder uses the attention weights to predict the next output in the sequence. Teacher forcing is used while training the decoder.

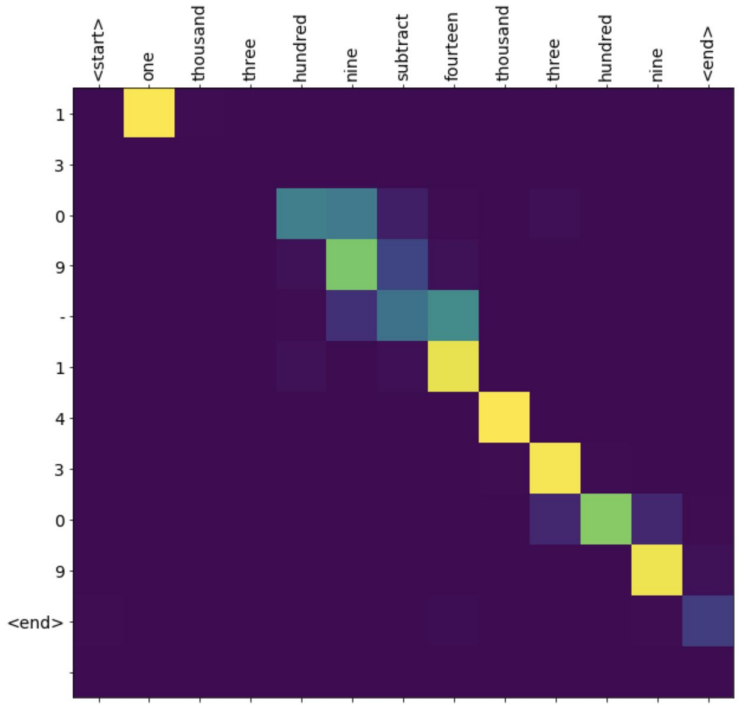


Source:
https://www.tensorflow.org/tutorials/text/nmt_with_attention#write_the_encoder_and_decoder_model

Results

Even though the dataset we have used is very large, because it is such a simple set of possible combinations, the model seems to be memorizing the entire dataset. This is evident from the below attention plots.

While the model validation accuracy after just 5 epochs is reaching 96.6, there are a few very simple cases such as ‘One plus Two’ where the model is not translating properly. This indicates that all the model is doing is memorizing the training data.



Single Variable Dataset

About the dataset

The dataset was created using a question generator for math word problems along with a list of ~2000 single variable equation questions from the Math Word Problem Repository (MAWPS).

In total, we have ~38000 question-equation pairs.

However, a caveat to mention here is that though the size of the dataset is quite large, the format of most problems is very similar (an issue with the question generator) and we were unable to find a more diverse dataset.

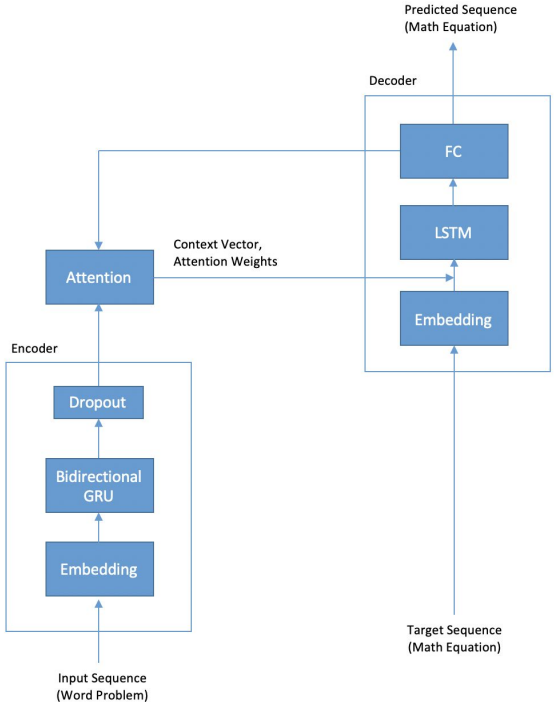
Seq2Seq(Bidir GL) Model Architecture

This model is very similar to the baseline model except of the following:

- The encoder has a bidirectional GRU layer
- A dropout layer is added to reduce overfitting
- LSTM layer is used in the decoder

The model does not memorize the dataset, giving us a reasonable BLEU score.

However, the attention plots are not as expected- not giving attention to correct parts of the sentence while translating.



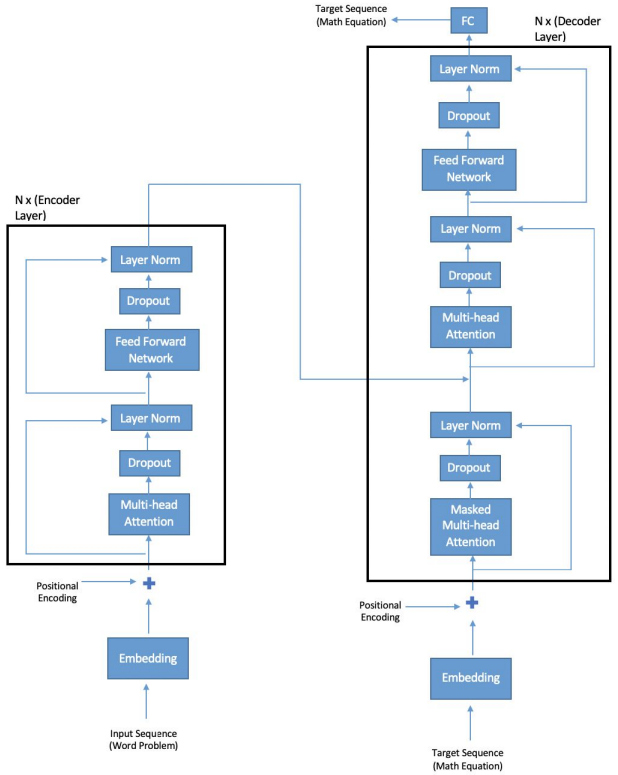
Source:
https://www.tensorflow.org/tutorials/text/nmt_with_attention#write_the_encoder_and_decoder_model

Transformer Model Architecture

The transformer model consists of the following blocks:

- An encoder, consisting of N encoder layers.
- Each encoder layer has a multi-head attention block and a feed-forward block
- A decoder, consisting of N decoder layers
- Each decoder layer has a masked multi-head attention block, a multi-head attention block and a feed-forward block
- Positional encoding is added to the input and target sequences since transformers have no recurrent units
- The output of the decoder then goes into a fully-connected layer which gives us our final prediction

We see that this model performs better than the Seq2Seq model, both in terms of the scores and the attention plots.



Source:
https://www.tensorflow.org/tutorials/text/transformer#top_of_page

Results

Comparison of the 3 models discussed above:

Model	**Seq2seq Baseline**	**Seq2seq Bidirectional**	**Transformer**
Corpus BLEU Score	0.9952	0.4478	0.7059
Accuracy	0.9656	0.2568	0.6860

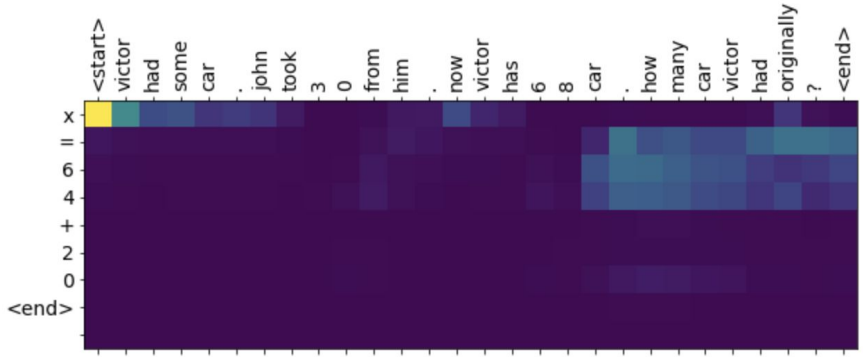
Results

While the seq2seq model gives a reasonable score on the validation set, when we look at the attention plots that are generated while translating a sentence, we see that attention is not given to the correct tokens when translating. This may indicate that this model is again starting to memorize the dataset rather than learning to actually convert a word problem to an equation.

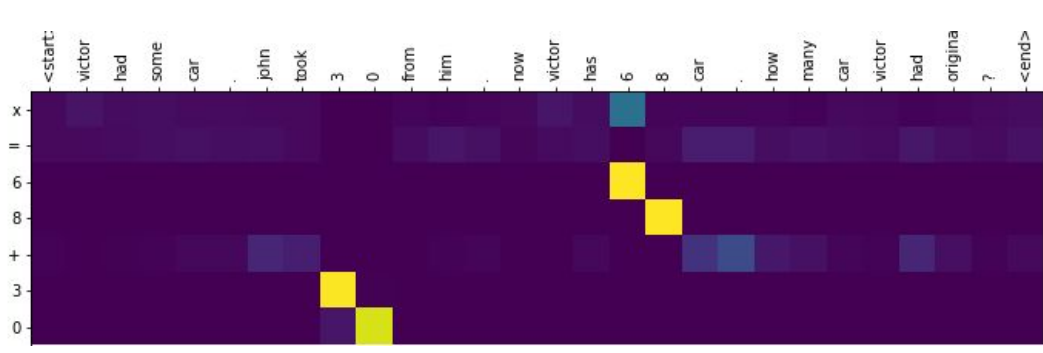
On the other hand, the Transformer model gives a higher score, and the attention plots also indicate that it is performing better than the seq2seq model.

Question: Victor had some car. John took 30 from him. Now victor has 68 car. How many car victor had originally?

Equation: X = 68 + 30



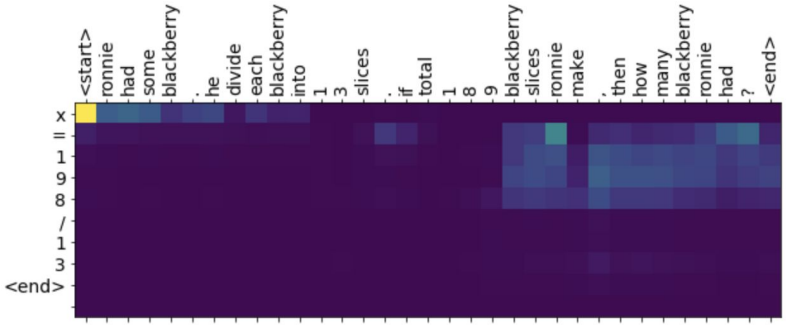
Bidirectional Seq2Seq



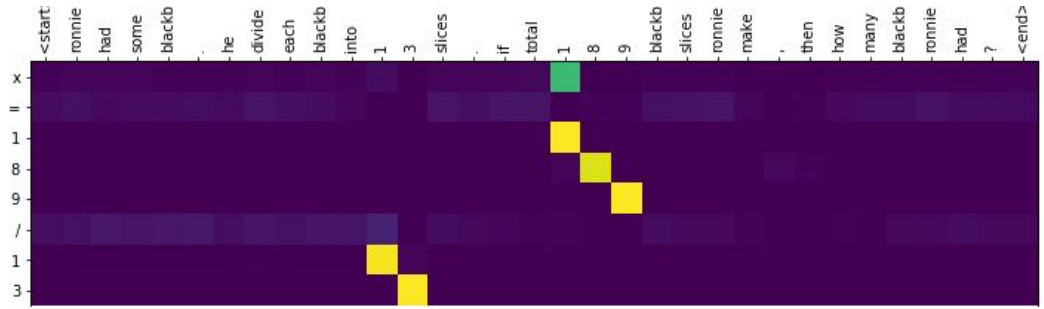
Transformer

Question: Ronnie had some blackberry. He divide each blackberry into 13 slices. If total 189 blackberry slices ronnie make, then how many blackberry ronnie had?

Equation: X = 189 / 13



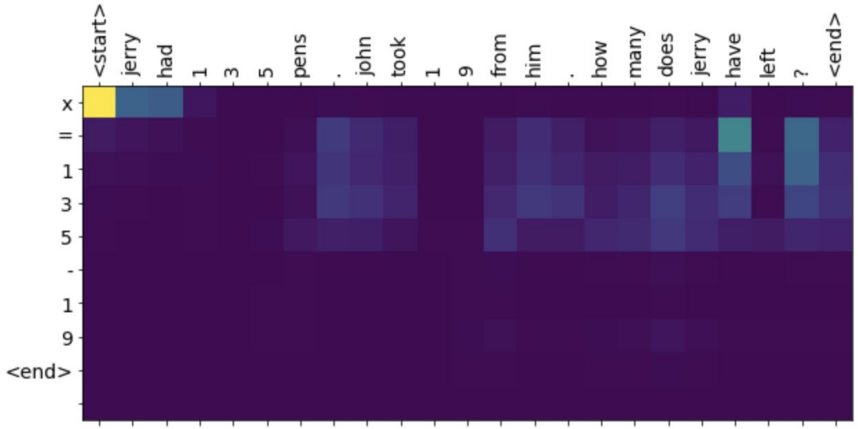
Bidirectional Seq2Seq



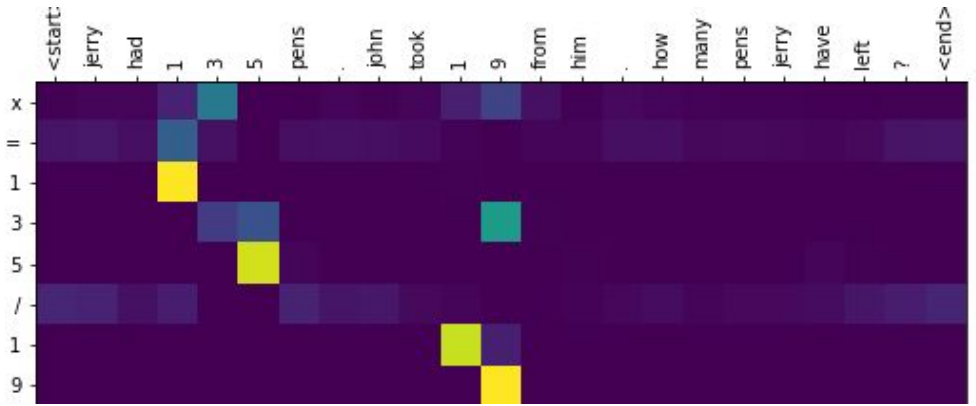
Transformer

Question: Jerry had 135 pens. John took 19 from him. How many pens Jerry have left?

Equation: X = 135 - 19



Bidirectional Seq2Seq



Transformer

Next Steps

- Further tune the model
- Get a larger and a more diverse dataset of word problems
- Experiment with multi-variable equations
- Handle the commutative and associative properties of math equations during prediction