

# Automatic Maths Word Problems

Supervisor: Dr. Anand Mishra

By: Anshu Priya  
Anupama Patel  
Megha Kumari

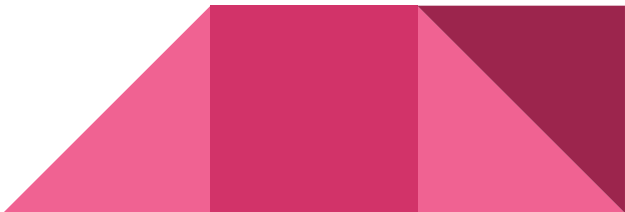
# Problem Statement:

Our aim with the project is to use Deep Learning to automatically convert simple math word problems.

A word problem is an exercise where significant background information is presented as text rather than mathematical notation. The description is followed by a pertinent question.

Word Problem
Oceanside Bike Rental Shop charges 17 dollars plus 7 dollars an hour for renting a bike. Tom paid 80 dollars to rent a bike. How many hours did he pay to have the bike checked out?
Equation
$17+(7*x)=80$
Solution
$x=9$

An example of arithmetic word problem.



# Datasets:

The dataset was created using “Question generator for math word problem” ([link](#)) and some are taken from “Math word problem Repository” ([link](#)).

In total, we have 3914 Question-Equation pair for single equation problem ([drive link for datasets](#)).

The number of problem in dataset is quite large but **most of the problem has kind of same structure as they are generated from question generator.**



# Tokenizing and Preprocessing datasets:

For the word problem, convert everything into lower case, add spaces around all punctuations and digits and removing any extra spaces.

While for equations convert everything into lower case and removing any extra spaces.

Tokenize the given list of strings and return the tokenized output along with the fitted tokenizer.

Add integers for start and end tokens for input as well as target expressions.

Pad all equation such that they are of equal length.

Text  
"The cat sat on the mat."  
↓  
Tokens  
"the", "cat", "sat", "on", "the", "mat", "."



# Baseline Model Architecture

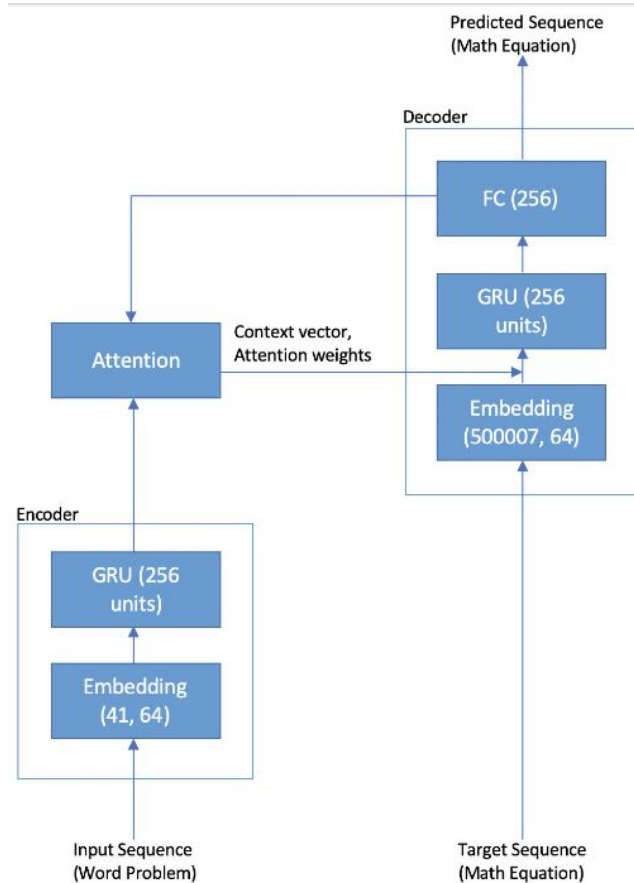
The encoder model takes in the word problem as a sequence.

The Bahdanau Attention computes attention weights for the input sequence along with the most recent prediction.

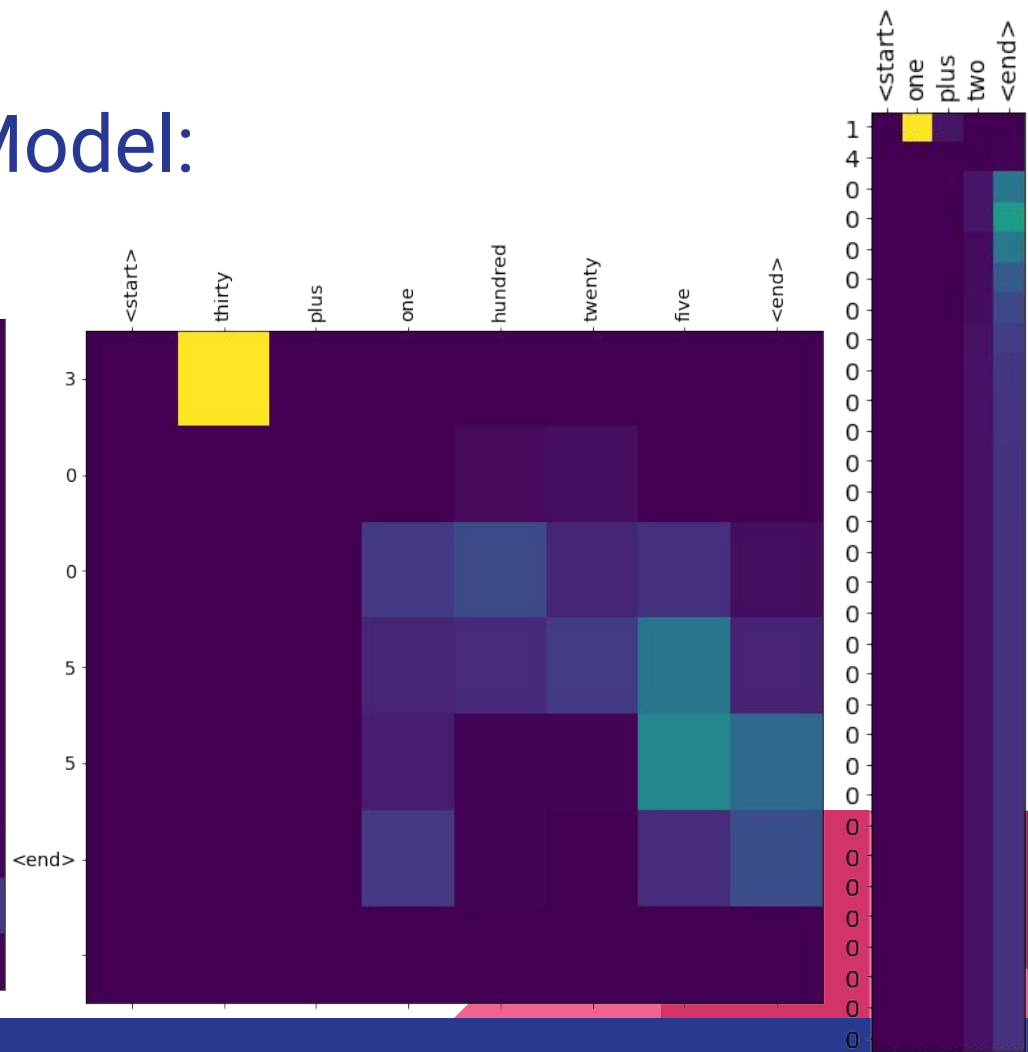
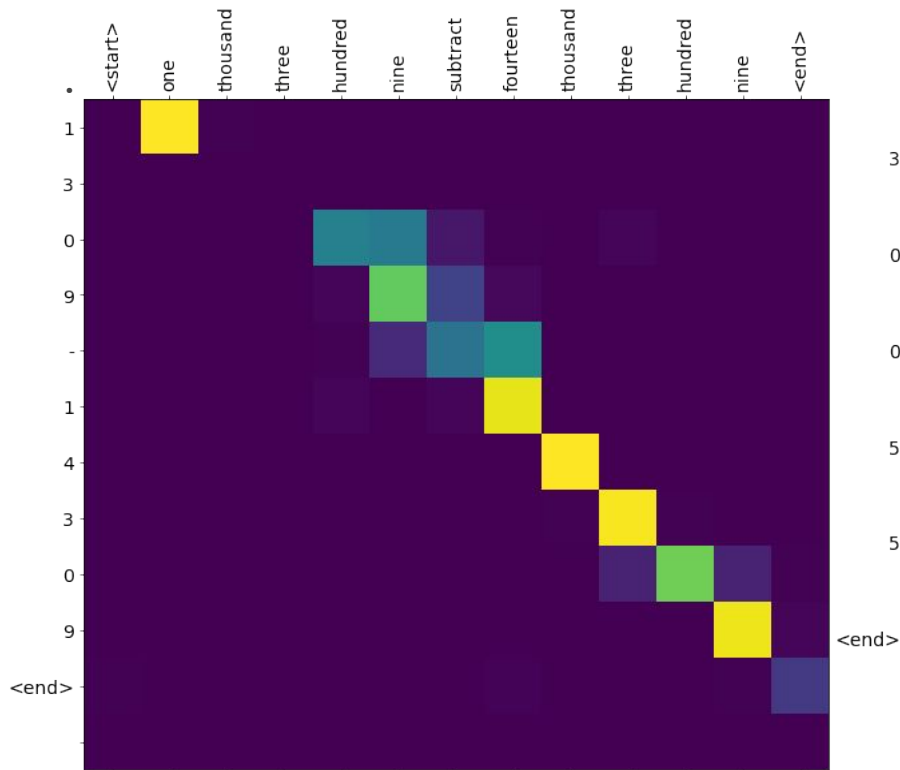
The decoder uses the attention weights to predict the next output in the sequence.

Teacher forcing is used while training the Decoder.

This model seems to be memorizing the entire dataset.



# Results on Baseline Model:



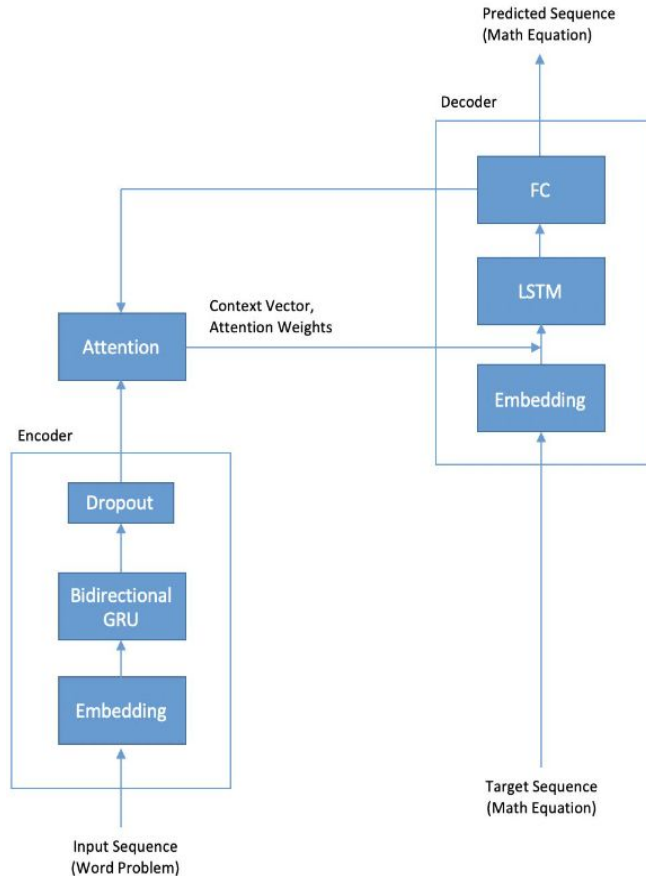
# Seq2Seq(Bidir GL) Model Architecture

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

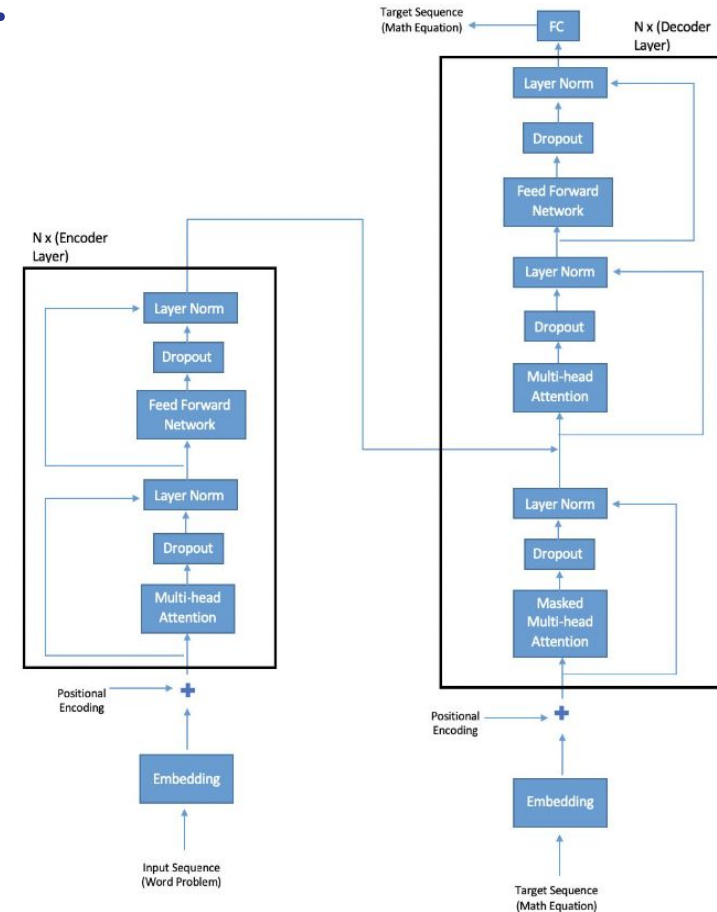
The model does not memorize the dataset, giving us a reasonable BLEU score.

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



# Transformer Model Architecture:

- 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






# Metrics used to calculate Accuracy:

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 mainly based on final result.

**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.



# Results:

Comparison of the 3 are given below:

Model	Seq2Seq Baseline	Seq2Seq Bidirectional	Transformer
Corpus BLEU Score	0.9952	0.4478	0.7666
Accuracy	0.9656	0.2568	0.5639

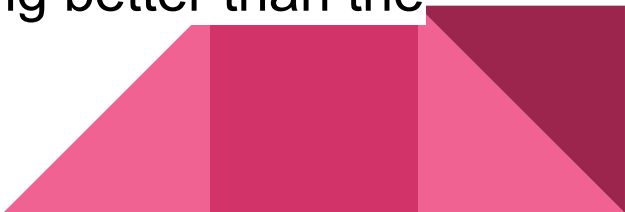


# Results:

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.

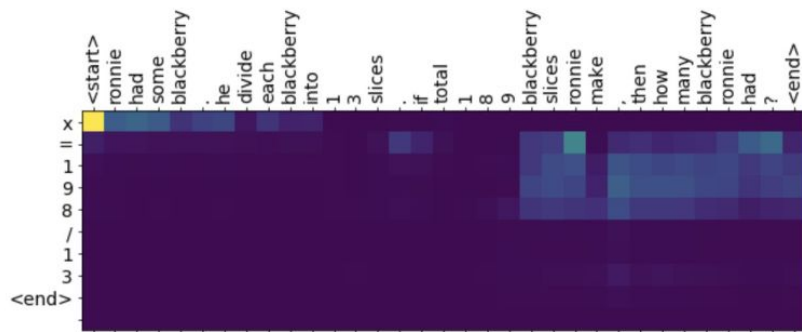
On the other hand, the Transformer model gives a higher score, and the attention plots also indicate that it is performing better than the seq2seq model.



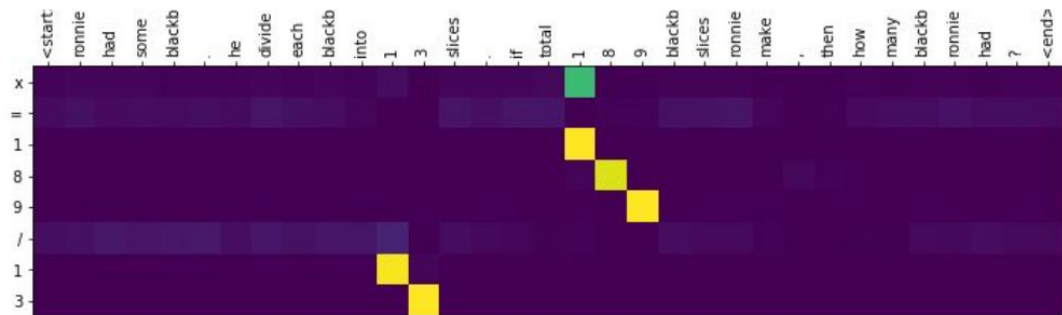
# Comparison between Seq2Seq and Transformer Model:

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

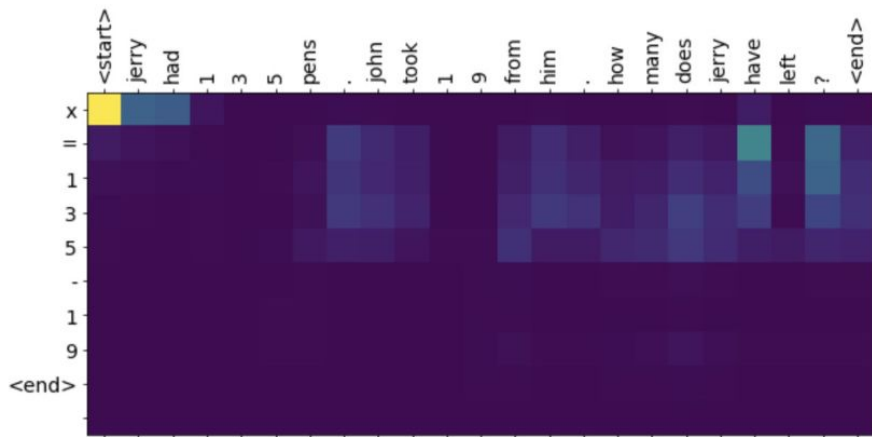


Transformer

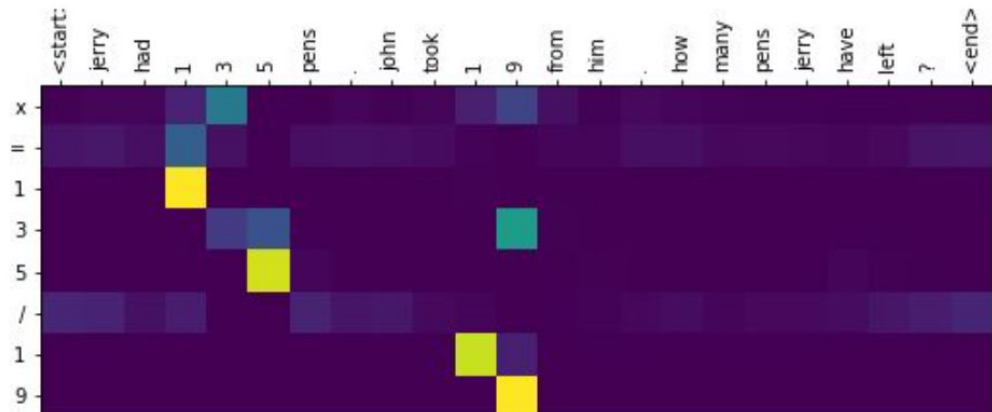
# Comparison between Seq2Seq and Transformer Model:

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

Equation:  $X = 135 - 19$



Bidirectional Seq2Seq



Transformer



# Thank You :)

We will be grateful if you provide any kind of suggestions to  
us :)