Fake News Classification

CS579 - Project II

Bharat Bandaru
Himamshu Lakkaraju

Objective

- Classify tweets into 3 categories (Agreed/Disagreed/Unrelated) based on two tweet titles available.

Dataset stats

Train Data: Total rows: 256,442

Data distribution:

Class	No of records	Percentage of data
Unrelated	175598	68.475
Agreed	74238	28.949
Disagreed	6606	2.576

As we can see from the above distribution of data, the classes are highly unbalanced and hence any classification approach most likely would not be able to perform or generalize well.

Methods to balance datasets

- Resample existing data.
- Generate new text similar to the existing text.
- subsample the train data and take similar amounts of data for each class.

Methods explored

- Generate new text similar to the existing text(GPT-2)
- **Use existing deep models to generate text:**

  There are several methods to generate text some of them include GPT2, BERT models etc. While these models are really good at generating grammatically correct sentences they did not always produce similar sentences as the given sentence (The method used is MASK words in sentence and obtain similar sentences)

- **Translate to another language and translate it back to english:**

  The other approach to create similar sentences is to translate text to another language using a ML model and translate it back to english. This would change some words in the sentences but would most likely preserve the overall meaning of the sentence with additional words added to the vocabulary.

- **Augment/replace words in text:**

  Replace word in text with synonyms.

Methods used

- **Translation of text to another language.**

  while this idea is good and it works technically, When we tried to use Helsinki models this took a long time for the whole dataset and hence this idea was discarded.

- **nlpaug synonym function:**

  This function generates similar words and replaces them in the sentences. This is quick and easy to generate text with similar meaning without any issues faced above and is used to generate data for agreed and disagreed class records. the minimum words to be replaced are set to 2 and the maximum words that can be replaced is set to 4.

Dataset stats after rebalancing.

Train Data: Total rows: 436,341

Data distribution:

Class	No of records	Percentage of data
Unrelated	175598	40.240
Agreed	148456	34.023
Disagreed	112287	25.734

Logistic Regression

A simple logistic Regression model is built with title1_en and title2_en combined. These combined titles are vectorized using the TF-IDF vectorizer and passed to the logistic regression model using data from the both titles to fit with minimum document frequency of around 20.

Logistic Regression

The train data is split into train and validation datasets (70%,30%)

Classification report for train data is:

	precision	recall	f1-score	support
0	0.85	0.87	0.86	122918
1	0.86	0.85	0.85	103919
2	0.96	0.93	0.95	78601
accuracy			0.88	305438
macro avg	0.89	0.88	0.89	305438
weighted avg	0.88	0.88	0.88	305438

Classification report for validation data is:

	precision	recall	f1-score	support
0	0.82	0.84	0.83	52680
1	0.82	0.82	0.82	44537
2	0.95	0.92	0.94	33686
accuracy			0.85	130903
macro avg	0.87	0.86	0.86	130903
weighted avg	0.85	0.85	0.85	130903

Advantages and Issues with Logistic Regression

Advantages:

- The model is very easy to build.
- The vectorizer fits the vocabulary of the training data so this would work well if the test data this is used on has similar patterns or vocabulary to the training dataset (which is not usually the case in the tweet/text data)
- The training time for the model is comparatively less than training a neural network.

Issues:

- Although the precision, recall and accuracy of the model is very high, this model is not a generalized model that would work for new data reliably as it's trained on the vocabulary of the available titles in the training data
- This also doesn't accurately depict the relation between the words in the titles or identify patterns that can be generalized.
- If the test data isn't similar to the training data, this model might not perform as well as it did on the training and validation data

Alternative models & Challenges

- There are several neural network architectures that work well for sequences like text like LTSM models or siamese nets that use LSTM layers or GRU layers to find similarities between the inputs or between inputs and a reference value.
- These nets usually capture the word patterns in the data far better than the logistic regression can.
- There are many existing models like BERT which can be used to classify the text with very high accuracies.

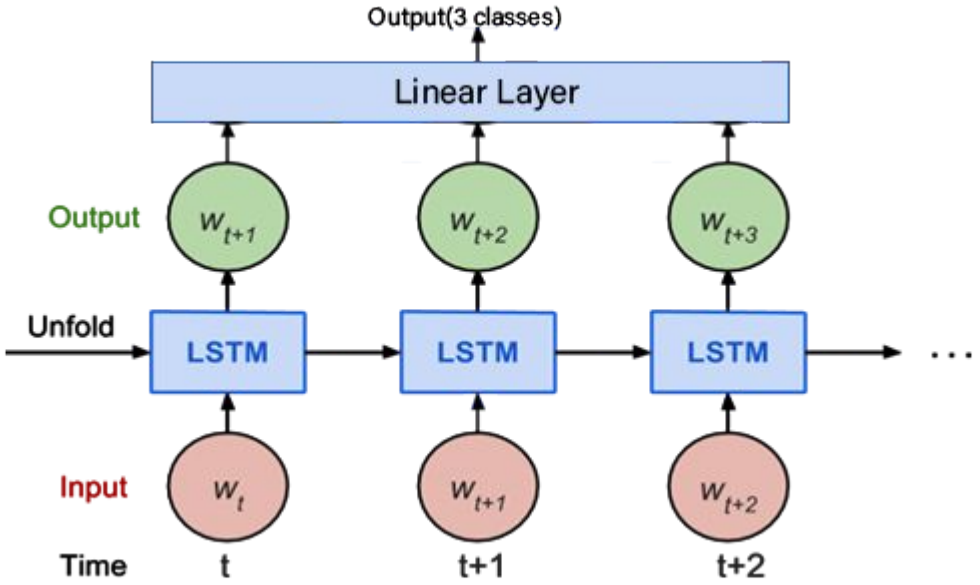
Challenges with neural networks:

- Neural networks tend to have overfitting/ underfitting issues
- Require large amounts of computing resources
- Hyper parameter optimization becomes difficult especially if the computational resources are limited.
- Have to train with multiple epochs so the training time will be longer.
- Imbalanced data would mean the network overfits to a class with large amount of data.

Simple LSTM neural Network.

- The title 1 and tile 2 are combined and are encoded.
- This is passed to a LSTM network with vocabulary size 49491, embed_dimension=256, lstm_units,hidden_dimension as 128 and 128.
- SGD optimizer with a learning rate of 0.01 and momentum of 0.9 is used.
- The loss function used is categorical cross entropy loss with weights calculated from sklearn class weights function to help the model generalize well.
- We could only use a batch size of upto 250 (due to cuda memory limitations)
- The loss goes into a plateau after an epoch and stays the same after. The accuracy is in the range of 34 to 50%.
- This might be primarily due to the batch size. With data this big the batch used heavily determines the loss and learning of the model.

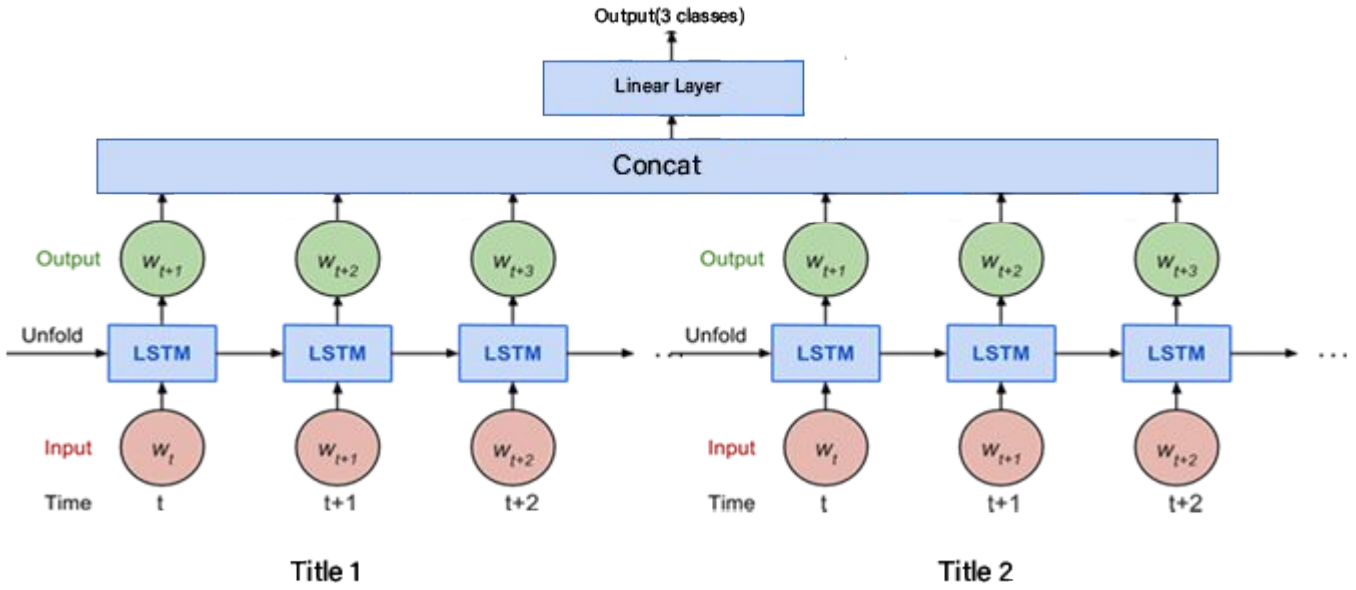
Simple LSTM architecture:



Siamese network:

- **These neural networks are used to identify similarities or dissimilarities between two input.**
- **The architecture is similar to the Simple LSTM model we created previously, it takes two encoded titles separately as inputs and passed through the LSTM. the outputs for each titles are concatenated and sent to the output layer.**

Siamese network:



Accuracy of LSTM and Siamese LSTM nets:

- The accuracy of our simple LSTM network is usually around 34%
- The accuracy of Siamese network improves accuracy a little. It goes up to 60%

Future Work:

- Since these models are just trained for 1 epoch and small batch size, we'd like to run this for more epochs with larger batch size and see if that improves the accuracy of these networks.

Thank You