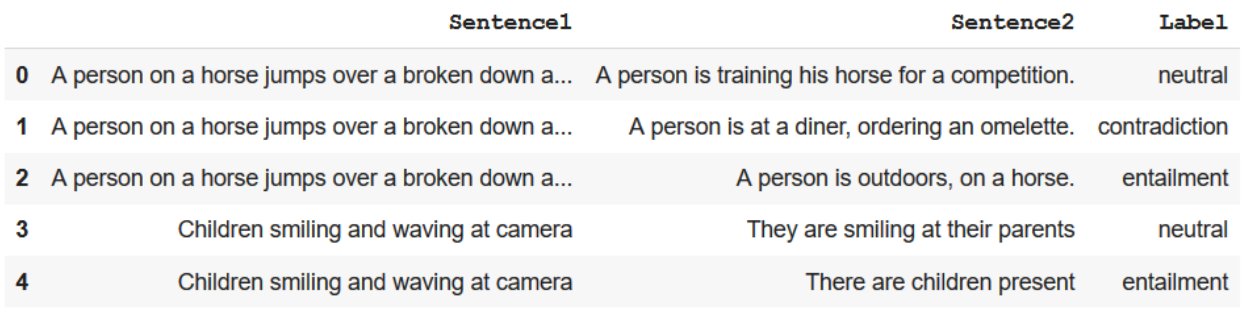
Deep Learning Assignment 3

NLP

Abin Bassam A

Task 1: TF-IDF based text classifier

Dataset: Here we used SNLI dataset with over 500,000 sentences and their labels. Here we have Text (Sentence 1) and Hypothesis (Sentence 2).Also their corresponding Label.



	Sentence1	Sentence2	Label
0	A person on a horse jumps over a broken down a...	A person is training his horse for a competition.	neutral
1	A person on a horse jumps over a broken down a...	A person is at a diner, ordering an omelette.	contradiction
2	A person on a horse jumps over a broken down a...	A person is outdoors, on a horse.	entailment
3	Children smiling and waving at camera	They are smiling at their parents	neutral
4	Children smiling and waving at camera	There are children present	entailment

Fig1. Sample of our Dataset

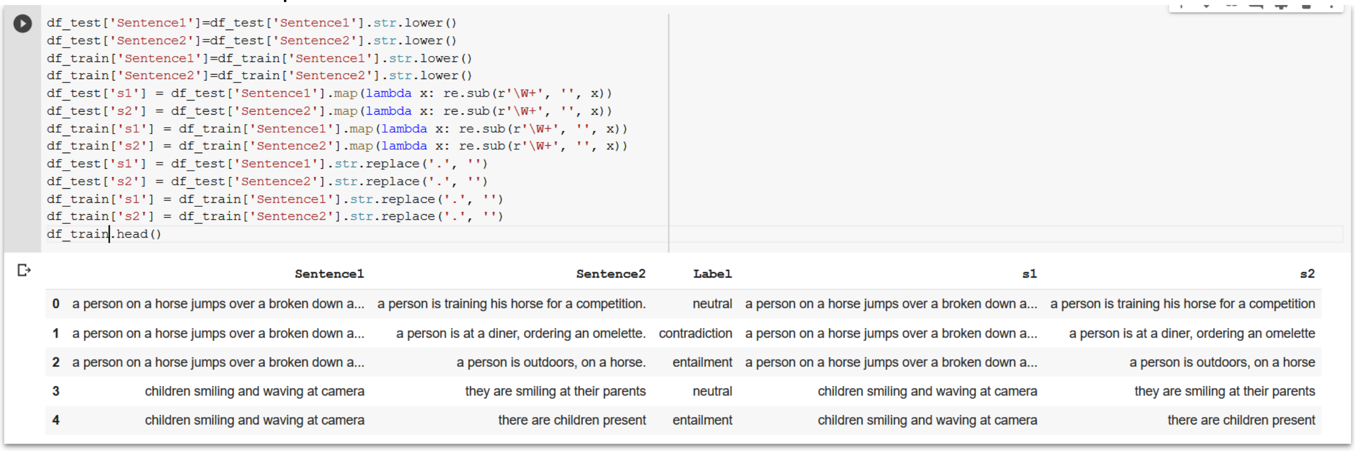
Since we can't pass the text as it is to ML algorithms, we need to vectorize the text data to numerical data.

Preprocessing and Vectorization

Raw data contain numerical value, punctuation, special character etc. these can adversely affect our vectorization and hence classification. So we need to preprocess the data before passing to the Vectorizer.

Here we did the following:

- All text to lower case
- Removed punctuations
- Removed Special Characters and Numbers

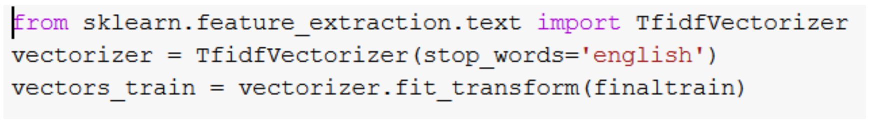


```
df_test['Sentence1']=df_test['Sentence1'].str.lower()
df_test['Sentence2']=df_test['Sentence2'].str.lower()
df_train['Sentence1']=df_train['Sentence1'].str.lower()
df_train['Sentence2']=df_train['Sentence2'].str.lower()
df_test['s1'] = df_test['Sentence1'].map(lambda x: re.sub(r'\W+', '', x))
df_test['s2'] = df_test['Sentence2'].map(lambda x: re.sub(r'\W+', '', x))
df_train['s1'] = df_train['Sentence1'].map(lambda x: re.sub(r'\W+', '', x))
df_train['s2'] = df_train['Sentence2'].map(lambda x: re.sub(r'\W+', '', x))
df_test['s1'] = df_test['Sentence1'].str.replace('.', '')
df_test['s2'] = df_test['Sentence2'].str.replace('.', '')
df_train['s1'] = df_train['Sentence1'].str.replace('.', '')
df_train['s2'] = df_train['Sentence2'].str.replace('.', '')
df_train.head()
```

	Sentence1	Sentence2	Label	s1	s2
0	a person on a horse jumps over a broken down a...	a person is training his horse for a competition.	neutral	a person on a horse jumps over a broken down a...	a person is training his horse for a competition
1	a person on a horse jumps over a broken down a...	a person is at a diner, ordering an omelette.	contradiction	a person on a horse jumps over a broken down a...	a person is at a diner, ordering an omelette
2	a person on a horse jumps over a broken down a...	a person is outdoors, on a horse.	entailment	a person on a horse jumps over a broken down a...	a person is outdoors, on a horse
3	children smiling and waving at camera	they are smiling at their parents	neutral	children smiling and waving at camera	they are smiling at their parents
4	children smiling and waving at camera	there are children present	entailment	children smiling and waving at camera	there are children present

**https://nlp.stanford.edu/pubs/snli_paper.pdf*

Then we passed it to inbuilt TF-IDF vectorizer in sklearn module. Our feature space was of the size **33241**



```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(stop_words='english')
vectors_train = vectorizer.fit_transform(finaltrain)
```

Fig 3. Tfidf Vectorizer. We also removed stop words from feature space

Logistic Regression

Now using the output of tf idf vectorizer as X and Gold Label as Y, we pass it to logistic regression and got the accuracy of 63.47% on the test data

Deep Learning Model

As we can see from above result, the accuracy of our classifier is below par. This may be due to bad vectorization as TF-IDF does not capture position in text, semantics, co-occurrences in different documents, etc and also limitation of linear regression on this high dimensional data (dim~=33K). So here we use **GloVe Embeddings** for better vectorization. The advantage of GloVe is that, unlike TF-IDF, GloVe does not rely just on local statistics (local context information of words), but incorporates global statistics (word co-occurrence) to obtain word vectors.

Here we use pretrained Glove Model from *Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors).*

Neural Network Architecture

Our Architecture is inspired from the paper "***A large annotated corpus for learning natural language inference***"*. We used RNN based architecture but here instead of tanh activation (as in the paper) layers we used Relu Activation.

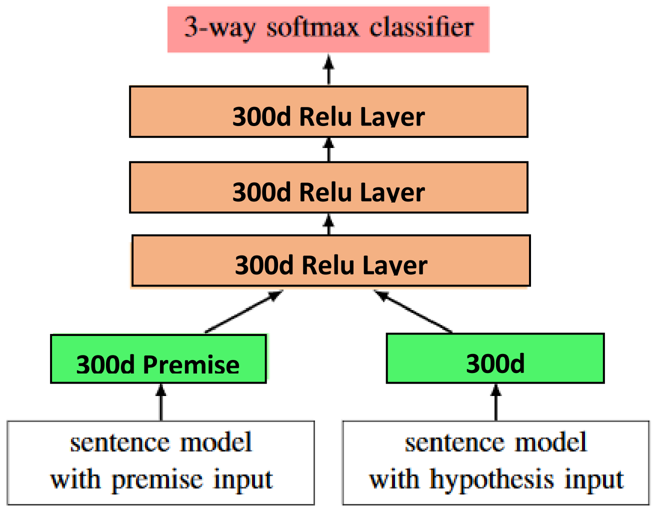


Fig 4. Network Architecture

**https://nlp.stanford.edu/pubs/snli_paper.pdf*

Results and Discussion:

With the Deep Learning Model our accuracy rose to 79.12%. Which is a significant increase from our TF-IDF Model. Varying size of hidden units didn’t affect the accuracy much. It was always hovering around 80%.

**https://nlp.stanford.edu/pubs/snli_paper.pdf*