Report – Interim Submission

Project: *Textual Coherence*

Team: *Advaith Malladi (2021114005), Patanjali B (2021114014), Revanth Gundam (2021114018)*

What was committed for Interim Submission:

- Dataset Finalization
- Problem Finalization
- Architecture Design Finalization

What has been completed for Interim Submission:

- Dataset Finalization (CNN Dataset)
- Problem Finalization
- Architecture Design
- **We build an RNN-LSTM based neural model to predict if the given paragraph is coherent or not: 96% accuracy and 0.96 F1 score achieved.**

What makes our mid submission so special:

- We have found an elegant way to represent each and every single paragraph with a matrix of size 300*4 such that no information is lost, the semantics are captured and such that the incoherence, if present, is evident.
- We have described this method in the section: **Paragraph to vector.**

Project Description:

- **Dataset:** For the dataset part, we have finalized the CNN corpus for discource coherence
- **Problem Finalization:** We have 3 sub-problems in our project
  - **Classification:** Classifying paragraphs as coherent or non-coherent
  - **Grading:** Assigning a coherence score to a paragraph in a small rangae (extension of previous task)
  - **Generation:** Given a paragraph, generating coherent sentences

What we have done so far:

- We have looked at different corpuses and finalized the CNN corpus for discourse coherence. This corpus contains approximately 73,000 coherent paragraphs out of which half serve as negative samples due to replacement of a single sentence from the paragraph at a random location with a random incoherent sentence
- ***We have created a RNN-LSTM based neural model which can understand paragraphs and predict whether they are coherent or not with 96% accuracy and 0.96 F1 score.***
- ***Now, we will go into the model design and architecture.***

NEXT PAGE FOR MODEL PERFORMANCE, ARCHITECTURE, ETC. --->

LSTM for discourse coherence classification (DONE):

Before going into details, here are the performance statistics of our model for motivation to keep reading :)

Performance Statistics:

Training:

- *average accuracy:* ***0.959704061109581***
- *average F1 score:* ***0.9592171332083778***
- *average recall:* ***0.9598704778547373***
- *average precision:* ***0.9598770763790266***

Testing:

- *average accuracy:* ***0.9530687785781591***
- *average F1 score:* ***0.9526936217142479***
- *average recall:* ***0.9530222501983382***
- *average precision:* ***0.9531950044037375***

Tasks in this model:

- Converting each paragraph into a vector of fixed dimensions to give as input to LSTM without loosing any information
- Choosing an architecture for my LSTM model
- Hyperparameter tuning for optimal results.

Dataset Nature:

- 73k paragraphs in total, each one tagged as coherent or incoherent.
- 50% of the paragraphs are coherent and 50% are incoherent.
- **There is a perfect split between coherent and incoherent, this makes our results even stronger and solid.**

Architecture and Hyperparameters for the model:

We first send our paragraph embeddings to the LSTM cell. From the last layer of the LSTM, the datat goes to the hidden2tag layer which takes in the data and gives out a distribution across the 2 tags: coherent and incoherent. Then the distribuiton is soft_maxed to give a probability distribution.

Loss function: negative log likelihood
optimizer: Adam optimizer
batch size: 128
number of epochs: 100
train:test split = 80:20
number of data samples = 73000 apprx

LSTM design:

- input size of first layer : 300
- hidden layer size : 300
- number of layers : 2

Hidden2tag design:

- input size : 4*(hidden layer size)
- output size : 2 (as we have 2 tags)

Paragraph to Vector (Paragraph Embeddings):

- We noticed that all paragraphs had a varying number of sentences and each sentence had a varying number of words. So, we had to find a way to represent each paragraph by a vector of fixed dimension which could be given as input to our neural model.
- Generating word embeddings during the training would also prove to be a herculean task as the task of predicting coherence itself is a complex task, generating embeddings during the training would not be fruitful as the global meanings of these words would not be captured and creating embeddings while training limits the semantics of the word to the given corpus.
- So, we decided to use GloVe Global Embeddings for our word embeddings. These are of dimension 300.
- Next, we decided to represent each sentence by the average of all the word embeddings of all the words in that sentence. Let Ei represent embedding of Si (sentence i):

```
#initiate Si to 0's, 300 dimension
for word in Si:
        Ei = Ei + word_embedding(word)
Ei = Ei/len(Si)
```

- Then, after we have sentence embeddings Ei's for all sentences Si's, we divide the paragraph into 4 almost equal parts on the basis of sentences, first 1/4th of sentences in first chunk, second 1/4th of sentences in second chunk and so on.
- Now, we have 4 chunks, we calculate the average sentence embeddings of these four chunks.
- So, now, we have four embedding vectors, each of shape 300*1, each one representing one chunk of the paragraph.
- Now, we conacatenate these four vectors horizontally to give a matrix of shape 300*4.
- Thus, we have successfully represented each and every paragraph as a matrix of shape 300*4, so this matrix serves as a kind of embedding for the paragraph.
- **Why this works:**
  - We know that incoherence is caused by one random sentence which is added to a random location of the paragraph.
  - This incoherent sentence must fall into any one of the chunks made in every paragraph.
  - Since the incoherent sentence differs in context, semantics, etc with all the other sentences of the paragraphs, the **embedding of the incoherent chuck will be quite different in value from the embeddings of all the other chunks. We hope our LSTM**

can identify the presence of the incoherent chunk given the context of the entire paragraph.

- The task of the LSTM model is to essentially detect a spike or dip in the chunk embeddings which might be caused due to an incoherent sentence.

Upcoming work and timeline:

- **by March 25th : Completion of Gradable model**, which assigns a coherency score to each paragraph
- **by April 6th: Completion of Generative model**, which generates coherent sentences given a coherent paragraph.
- **By April 9th : End of testing** and final submission

--THE END--