Text Coherence Analysis on Social Media

Aman Raj (006) **Tushar Bhutada (073)** **Udbhav Chugh (081)** **Mayank Baranwal (084)**
Indian Institute of Technology Guwahati
Group 8, Mid-Term Report
{aman170101006, tusha170101073, udbha170123055, baran170102035}@iitg.ac.in

Abstract

Text coherence is a necessary, but often overlooked aspect of communication on social media. The unique nature of social media such as unstructured text, makes it challenging to analyze text coherency. In the mid-term report, we cover the methodology utilized for the automatic generation of coherent and incoherent datasets. Furthermore, we describe the model input generation pipeline. Finally, we highlight the progress made in these aspects and the remainder of the work to be done.

1 Introduction

Coherence is a sub-domain of linguistics that covers the features of semantically meaningful text. Coherent passages have smooth transitions and an easily understandable overall theme. The different thoughts should flow evenly from one paragraph to the next. Ensuring coherence is key to enabling easy comprehension of an idea. A lack of coherence can convolute the most simple concept beyond comprehension. Thus, coherence is key to ensuring effective communication, immaterial of the platform.

1.1 Coherence in Social Media

Social media is a platform that provides the smallest of voices with an unparalleled reach: clear messaging has the potential to be heard across the world. Though a majority of users write sensible sentences, an argument is perceived as coherent only if the ideas flow smoothly from one paragraph to another. Furthermore, users must ensure that there isn't an unreasonable drift from the original topic.

More often than not, social media users start a discussion on a particular topic but finish on an entirely different theme. Consequently, the post may not effectively convey the intended message. Worse still, malicious actors may choose to obscure the facts by convoluting a topic beyond comprehension. Thus, it is necessary for social media users to not only be coherent but also have the capability to recognise an incoherent argument.

To analyse and overcome these issues, we aimed to generate new datasets that reflect the conversational nature of social networking discussions. Furthermore, we looked at implementing a novel architecture for better assessment of user post coherency. We now discuss the existing literature, our method and progress and future tasks planned in this project.

2 Related Work

2.1 Available Datasets

For coherence tasks, two corpora are widely used for text coherence analysis(Barzilay and Lapata, 2005). The first is a collection of National Transportation Safety Board's aviation accident reports and the second contains Associated Press articles from the North American News Corpus on the topic of earthquakes. Both the dataset consists of ordered pairs of alternative renderings (x_{ij}, x_{ik}) of the same document d_i, where x_{ij} exhibits a higher degree of coherence than x_{ik}. Existing datasets are synthetically generated using an original document (x_{ij}), taking its different permutations and forming the pair (x_{ij}, x_{ik}) with x_{ij} being the original document (and is more coherent), for each permutation.

2.2 Text Coherence Models

A variety of approaches have utilised taken up to analyse the problem of text coherence. (Barzilay and Lapata, 2005) created an entity-grid representation of documents, which captures patterns of entity distribution in a text like distributional, syntactic, and referential information. (Elsner and Charniak, 2008) developed a discourse-new classifier and pro-

noun coreference method for the information ordering task in text coherence analysis. (Mesgar and Strube, 2016) implemented a lexical coherence graph to represent lexical relations among sentences and used them to describe the connectivity style of sentences in the document. It connects sentences based on semantic relations between words. (Li and Jurafsky, 2017) used neural models to measure multiple aspects of coherence in existing sentences and also maintain coherence while generating new sentences. Their work included coherence analysis and coherent text generation. (Mesgar and Strube, 2018) used recurrent neural networks and LSTM for computing semantic vectors for sentences. Here, the information in word embeddings and the surrounding context is combined using recurrent networks (RNN). (Cui et al., 2017) used a convolution neural network (CNN) architecture to capture text coherence. They also calculated the similarities of their distributional representation to capture the interactions between sentences and analyse coherence using a CNN architecture.

3 Task Description

The principal goals of this project are two-fold. Firstly, we aim to generate a new dataset that can be used to train models for text coherence analysis in a social media context. Secondly, we want to provide a novel model architecture for coherence analysis on social media text and analyse its performance on text coherence applications in general. Thus far, we have completed the datasets generation task, cleaning and pre-processing of data. Also, we have implemented the entity extraction and corresponding word embeddings generation process. The details of our method and progress are described in detail in the following two sections.

4 Method and Progress Part 1: Datasets

4.1 Requirement of Separate Datasets

Social Media posts and texts vary significantly from other forms of texts. The tone is often informal and is often accompanied by heavy usage of slang and short forms which do not occur in regular corpora. Colloquialisms also make it hard to identify and classify keywords and entities. Furthermore, in many social media posts, users will start a discussion on a particular topic but finish on a tangential note. Finally, there is a considerable variation in themes and a lack of consistent structure in many posts. Hence, it is essential to have a corpus that reflects the nature of social media if we want the model to learn the appropriate characteristics.

4.2 Dataset Characteristics

Taking into consideration the valuable feedback from our professors, we decided to generate two different datasets. Of these, one has the posts and comments as separate documents (as planned earlier). The other one had posts and comments concatenated to have a better flavour of discussion on a topic as suggested by sir. We kept a minimum character limit of 1000 for final documents in both datasets to capture at least 10-12 lines per document.

Out of all the popular social media platforms, after analysing the posts and discussions on them, we chose Reddit because of its varied types of conversations and no character limit. This provided us with different post lengths, discussions and comments on a variety of topics and enabled us to create a dataset which can capture social media discussions of a wide variety. We used Reddit Top 1000 posts dataset provided publicly on Kaggle to generate our synthetic dataset.

From the Reddit Top 1000 posts dataset, we chose four different subreddit topics, namely Books, Explain like I'm five, AskReddit and Jokes to have that variety in the dataset. We extracted an equal number of documents from each subreddit to keep a consistent yet broad dataset. The exact details are provided in subsection 4.3.

While concatenating comments to a particular post, we had the option of choosing between the top comments or an author's replies on the post. We decided to take the Top 10 upvoted comments in the parent thread on each post and concatenated them to the post. This is because an author's replies to some comment in any child thread need not necessarily be an extension to the post. It may even go a bit off the topic as it will be more focused on replying to a particular comment. As such, the top 10 comments are more relevant to the post's theme and are thus considered.

4.3 Dataset Extraction and Generation

For the data extraction, we have used the Selenium tool in Python, a well-known open-source framework developed for such extraction tasks. To install and configure its Chrome webdriver in Google Colab, we used a small library named Kora.

Algorithm 1 Dataset generation

Input: $\mathcal{RD}$ – Reddit Top 1000 posts dataset
Output: $\mathcal{D}$ – Dataset

function GENERATEDATASET
  $requiredDocs \leftarrow 70$
  $limit \leftarrow 1000$
  $permutCount \leftarrow 5$
  for $subreddit \in \mathcal{RD}$ **do**
    $urls \leftarrow list(subreddit)$
    $found \leftarrow 0$
    for $url \in urls$ **do**
      $post \leftarrow getPost(url)$
      $comts \leftarrow getComments(url)$
      $doc \leftarrow post$
      for $comt \in comts[0:10]$ **do**
        $doc \leftarrow doc + comt$
      end for
      if $charCount(doc) \geq limit$ **then**
        $found \leftarrow found + 1$
        $sents \leftarrow sentTokenize(doc)$
        for $k \leftarrow 1$ to $permtCount$ **do**
          $new \leftarrow shuffle(sents)$
          $\mathcal{D} \leftarrow \mathcal{D} + pair(doc, new)$
        end for
      end if
      if $found = requiredDocs$ **then**
        break
      end if
    end for
  end for
  return $\mathcal{D}$
end function

Using Selenium, we navigated through web URLs provided in the Reddit Top 1000 Dataset. Since every post has the same fundamental layout, we were able to detect static patterns in their HTML source of the webpage. This enabled our web-scraper to extract the necessary pieces of information and save it to a file or a database. Specifically, the static patterns were the *data-click-id* and *data-test-id* attribute of the HTML divs containing the Reddit post and its comments respectively. We used methods from Selenium like $find\_elements\_by\_xpath$ to get those elements and obtain their data.

With this, we extracted the coherent part of our dataset. Similar to (Barzilay and Lapata, 2005), we generated pairs of documents with one document more coherent than the other. To achieve this, we labelled the original post as coherent and the permutations of its sentences as incoherent. The underlying assumption is that the original sentence order in the source document must be more coherent than its permutation. Since the coherence relation between different permutations is unknown, our corpus includes only pairwise rankings that comprise the original document and one of its permutations. For permutation generation, we first broke down our document, which is in the form of a paragraph into sentences, for which we used the $sent\_tokenize$ method of the NLTK tool. We took 70 documents from each subreddit having a minimum of 1000 characters each to ensure we have over 10-12 lines for each document to analyse text coherence appropriately. We generated 5 random permutations of sentences of each document resulting in 70*5 = 350 documents per subreddit. Hence, we generated a total of 4*350 = 1400 pairwise records for training and testing. This process is repeated for both the datasets: one in which the posts and comments are considered separate documents and another where posts and top 10 comments are considered together.

Algorithm 1 shows the steps for dataset generation of type 2 dataset where post and top 10 comments are concatenated together. For type 1 dataset, the only difference is that instead of concatenating posts and comments together, they are checked separately for character count greater than thousand and are added in dataset as separate documents. This approach of dataset generation for social media texts is similar to what has been used in standard text coherence analysis applications, as explained before.

5 Method and Progress Part 2: Coherence Model

Having extracted the dataset, we now need to refine it for model training. This step is especially critical for social media texts as they tend to use a large number of colloquialisms. We now detail the pipeline specified in Figure 1.

5.1 Cleaning up the Dataset

We created the following pipeline for data preprocessing:

1. **Sentence Level Cleaning**: The first step is to extract the sentences from each post. We do this with the aid of NLTK's $sent\_tokenize$ package. Then, we proceed to remove emails, newline characters and single quotes in the

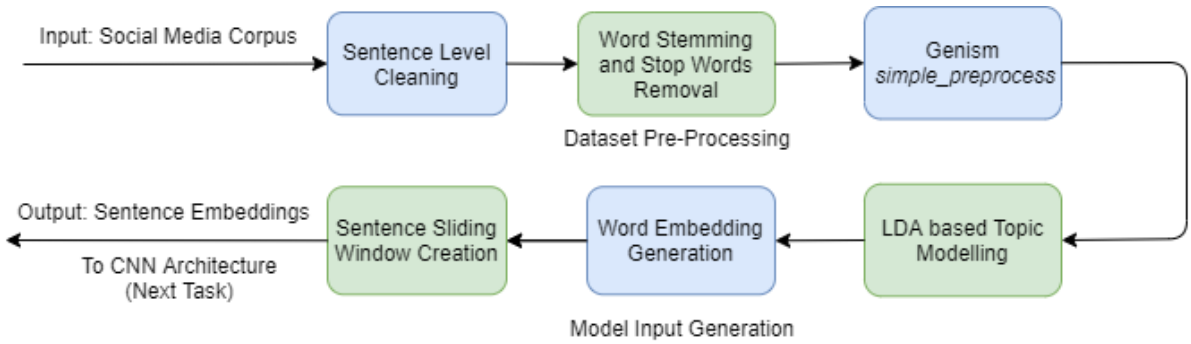

Figure 1: Pipeline for Coherence Model

individual sentences. With this, we remove all non-alphanumeric characters from the text. The removal of these characters ensures that a model does not recognize the same entity as separate ones due to erroneous punctuation. It also allows us to remove nouns (such as emails, hashtags etc) that do not significantly contribute to text coherence but can serve as red-herrings for coherency evaluation.

2. **Word/Token Level Cleaning**: We stem each word with NLTK's $PorterStemmer$ package and stores it's root instead. Each word is also passed through a set of pre-defined filters to remove conjugating and stop words. Finally, we include only stemmed words which fall under the category of noun, adjective, verb or adverb.

   In posts on social media, punctuation and grammar may not always be appropriately utilised. Stemming words and extracting nouns, adjectives etc allows a model to accommodate a variety of writing styles. It also ensures that during training, the model is not chasing a "moving goalpost" due to variance in writing formats.

Additionally, the data is also passed through standard pre-processing models like genism's $simple\_preprocess$ and others. These deal with more peculiar character level modifications like lowercasing, de-accentations etc. It also tokenizes the sentences again (corpus merged prior to running this) with attention to punctuation.

At the end of this stage, we have the relevant entities that can be used for inferring text coherency. However, there can still exist multiple entities that can mislead the model in detecting coherency. Consequently, we must address these issues before generating embeddings.

5.2 Extracting Relevant Entities

As mentioned previously, certain entities can unnecessarily detract from the model's coherency analysis, even if the text is coherent. To counter such entities, we turn to topic modelling with Latent Dirichlet Allocation (LDA) (Jelodar). With the aid of LDA, we generate some of the dominating themes in the text. Note that these themes are statistically developed and are thus arbitrary from a linguistic standpoint. Each topic has a weighted contribution from different words. We then select the entities that match these topics and use them for generating word embeddings. As we reorder the sentences to generate incoherent corpus pairs, we note that the entities extracted are same and it is their order that helps distinguish the incoherency.

5.3 Generating Word Embeddings

To allow the model to make sense of the words, we generate their respective word-embeddings. For this purpose, we employ gensim's partially trained $Word2Vec$ model. We train the model on a separate, pre-existing dataset that is an agglomeration of new articles, tweets and more formal social media posts. Then, we pass the pre-processed entities from our corpus and obtain their embeddings.

Utilizing up a more formal dataset of words ensures that we can avoid having to process colloquialisms in the actual model. Slang words can have multiple interpretations and hidden meanings, making them difficult for models to comprehend. However, applying this constraint by itself has the potential to neglect or reject a large number of entities that may be relevant. As such, we convert as many colloquialisms to their root words as possible in the previous stage. Such a conversion may sacrifice some of the more subtle nuances of the corpus but allows us to employ more standard models for text coherence evaluation. Also, not removing col-

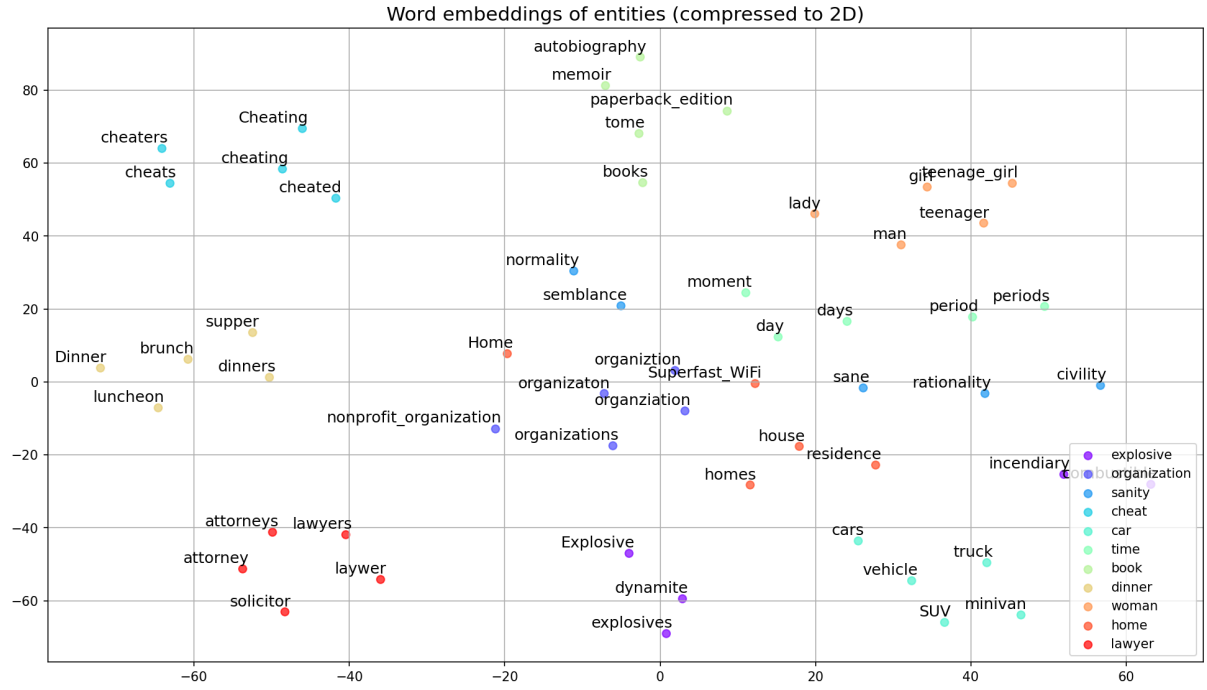

Figure 2: Generated Word Embeddings Clustered by Topic

loquialisms can result in multiple entities with the same meaning, which can confuse the model's evaluation process.

To aid in visualization, we show some of the resulting word embeddings, compressed to 2 dimensions in figure 2. We observe that the entities of a similar topic (defined by legend specifying the dominant keyword) are typically clustered together, as expected.

6 Links for Source Code and Resources

The following are the links to source code and generated datasets where outputs described in the paper can be seen:

- Coherence Dataset with posts and comments as separate documents.
- Coherence Dataset with posts and comments concatenated.
- Colab Notebook for dataset extraction and generation.
- Colab Notebook for preprocessing and model architecture.

7 Conclusion and Future Work

As planned initially, we have completed the dataset generation part, and now have two distinct datasets for text coherence analysis. From the model architectural point of view, we have completed the necessary preprocessing and cleaning of data keeping the nature of the dataset in mind. Further, we have added the model to extract entities from sentences and form word embeddings matrix.

Now, we plan to use this word embeddings as an input to a neural network architecture which we will train on the two generated datasets and analyse the results. As per the train and test split of data, we will keep five posts from each of the four subreddits present in our datasets. For each such post, we have generated 5 permutations of text, hence out of the 1400 data points we have; we will be using 1300 for training and 100 for testing.

As per the architecture, we have decided to pursue an approach similar to the one followed in (Cui et al., 2017). We will use a sliding window approach along with the already implemented entity extraction and word embeddings generation. Thus, we will create a 2D matrix for each sentence in the window (word x embeddings). Formally, if there are d features in a word embedding and there are s words in a sentence, we will have a d x s matrix for this sentence representing its word embeddings. Taking such matrices for each sentence in the window, we will use a CNN architecture to train the parameters. The exact layers of architecture will be finalised based on testing and what gives the best results. The overall structure of the CNN architecture would be such that it takes the d x s matrix

of different sentences present in a window, applies convolution and pooling layers and finally flattens it to generate a coherence score. As mentioned in (Li and Hovy, 2014), the coherence score of the entire text will be the product of coherence scores of each window of sentences. Overall, in a comparison between two documents, the document with higher coherence score will be considered more coherent than the other.

8 Individual Contribution

For the dataset generation part, Aman and Tushar wrote the code for extraction of posts and comments using the post links provided in the Kaggle's Top 1000 dataset. Udbhav and Mayank implemented the program for generating permutations for each document and storing the final results as a .tsv data file.

For the coherence model part, Aman and Tushar implemented the initial dataset loading, tokenization and later generating word embeddings from entities. Udbhav and Mayank wrote the code for remaining preprocessing like stemming, lemmatization, etc. and used this preprocessed data to extract topic-based entities using LDA.

The remaining part of the architecture (to be implemented next) was discussed by everyone based on different existing approaches and then finalised.

Acknowledgments

We would like to acknowledge Dr. Ashish Anand and Dr. Amit Awekar for their valuable feedback, teaching and guidance throughout the course.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, page 141–148, USA. Association for Computational Linguistics.

Baiyun Cui, Yingming Li, Yaqing Zhang, and Zhongfei Zhang. 2017. Text coherence analysis based on deep neural network. CIKM '17, page 2027–2030, New York, NY, USA. Association for Computing Machinery.

Micha Elsner and Eugene Charniak. 2008. Coreference-inspired coherence modeling. In *Proceedings of ACL-08: HLT, Short Papers*, pages 41–44, Columbus, Ohio. Association for Computational Linguistics.

Yuan Feng Jelodar, Wang. Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2039–2048, Doha, Qatar. Association for Computational Linguistics.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 198–209, Copenhagen, Denmark. Association for Computational Linguistics.

Mohsen Mesgar and Michael Strube. 2016. Lexical coherence graph modeling using word embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1414–1423, San Diego, California. Association for Computational Linguistics.

Mohsen Mesgar and Michael Strube. 2018. A neural local coherence model for text quality assessment. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4328–4339, Brussels, Belgium. Association for Computational Linguistics.