TOPOLOGICAL REPRESENTATION OF TEXT FOR ENTAILMENT

by

Ketki Savle

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Information Technology

Charlotte

2019

Approved by:

_____
Dr. Minwoo Lee

_____
Dr. Wlodek Zadrozny

_____
Dr. Samira Shaikh

ABSTRACT

KETKI SAVLE. Experiments in Text Entailment using various feature selection.
(Under the guidance of Dr. MINWOO LEE)

Inference has been a central topic in artificial intelligence from the start, but comparatively little progress has been made on the problem *natural language inference* (NLI), that is, determining whether a text can be inferred from another text. Language inference is at core of multiple NLP tasks such as information retrieval (IR), question answers (Q&A), machine translation and text summarization by improving the quality of machine's language understanding. For stronger reasoning and developing machine's understanding of natural language, we establish text entailment.

Stanford Natural Language Inference (SNLI) [Bowman et al. (2015)] and Multi-Natural Language Inference (MNLI) [Bowman et al. (2017)] are two of the standard datasets used for training the models for establishing text entailment. The major limitation of these two datasets is, when machine learning classifier or neural network models are trained on them, they cannot generalize very well on specialized tasks such as, legal data, dialogue systems and larger documents etc. In this thesis, we focused on establishing entailment for legal domain using topological representation of data. To proceed with our aim, we participated in a Competition of Legal Information Extraction and Entailment (**COLIEE** 2018), organized by University of Alberta, Canada. We also verified our models on SNLI and MNLI datasets to discuss the effectiveness and limitations of our approach.

ACKNOWLEDGEMENT

Firstly, I would like to thank my advisor, Dr. Lee, for giving me the opportunity, guidance and support to complete my thesis in one of the most interesting areas of Natural Language Processing. He is one of the most knowledgeable and versatile mentors who let me explore different ideas while steering me in the right direction. I am truly grateful to Dr. Lee for encouraging me to never give up, no matter how hard the problem is! This wouldn't have been possible without a teacher like you who always pushed me out of my comfort zone to achieve excellence.

I would like to express my gratitude to my committee Dr. Samira Sheikh and Dr. Wlodek Zadrozny for being a solid support throughout the journey and helping me overcome the most difficult challenges in terms of knowledge gap and implementation. Their ideas and suggestions helped me accomplish my thesis successfully. I am very grateful to Dr. Zadrozny for being an amazing teacher from the beginning of my journey in the program.

I am also thankful to College of Computing and Informatics for providing the necessary resources for computation while working on this research. This wouldn't have been possible without your support.

This would not have been possible without a strong support of my husband Abhijeet and our family. I thank you all for always trusting my opinion and decisions while we accomplish my masters program. Last but not the least my fellow researchers and PhD colleagues who always guided me with their excellent knowledge and experience which kept this work aligned with the end goal. A big thank you to you all!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

Adj	Adjacency
BM25	Best Match 25
COLIEE	Competition of Legal Information Extraction and Entailment
CNN	Convolutional Neural Network
LSTM	Long-Short Term Memory
MNLI	Multi-Natural Language Inference
NLTK	Natural Language Tool Kit
NLP	Natural Language Processing
NLI	Natural Language Inference
SNLI	Stanford Natural Language Inference
TE	Text Entailment
TDA	Topological Data Analysis
TFIDF	Term Frequency Inverse Document Frequency
XML	Extended Markup Language

CHAPTER 1: INTRODUCTION

In Natural Language Processing, the most critical part for machines is to well understand the fundamental meaning of natural language. Given a set of sentences or words, machines should be able to understand the meaning of those two words correctly. Such understanding plays a critical role when it comes to retrieving most semantically relevant documents for user's search query. Also, for answering the questions the answer is not direct and requires inference of context to provide logical reasoning. Machines cannot think on their own, thus, to achieve higher results for language tasks, establishing semantic and syntactic relationship plays a major role in natural language processing.

1.1 What is Text Entailment?

Text Entailment (TE) or Natural Language Inference (NLI) in natural language processing refers to determining a directional relation between two text fragments. The relation holds whenever one sentence can be inferred from another. More formally, a text *t* is said to entail a textual hypothesis *h* if the truth of *h* can be inferred from *t*. [Degan et al. (2005)] defined text entailment as

> "a text *t* entails a hypothesis *h* if, typically, a human reading *t* would infer that *h* is most likely true."

According to the [Kouylekov and Magnini, (2005)], *t* entails *h* if we have a sequence of transformations applied to *t* such that we can obtain *h* with an overall cost below a certain threshold, empirically estimated on the training data.

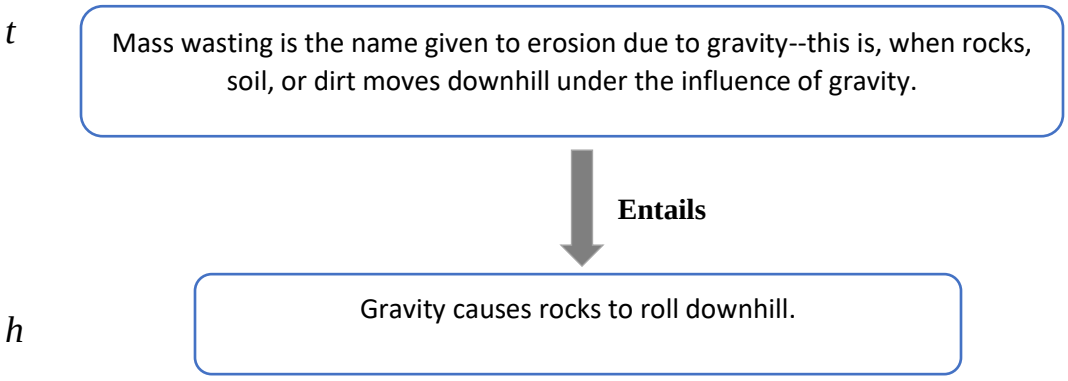

Figure 1-1 Example taken from SciTail dataset which shows the entailment relationship between text t and hypothesis h.

Figure 1.1 shows an example entailment from SciTail [Khot et al. (2018)] dataset. This example holds true for entailment because one can infer the fact, gravity causes rocks to roll downhill as supporting text *t* states rocks, soil and dirt moves downhill due to gravity. Thus, hypothesis follows premise and these two statements are entailed with each other. Information about presence or absence of entailment between two sentences has been found to be beneficial for the range of NLP tasks such as Question and Answering [Dagan et al., (2006); Harabagiu and Hickl, (2006)] and machine translation [Saikh et al., (2017)]. In natural language tasks, integration of more syntactic and semantic knowledge can yield superior performance.

1.2 Problem Statement

Typically, entailment can be established as a classification task. This could be a binary classification or a three-class classification task. Stanford Natural Language Inference (SNLI) and Multi-Natural Language Inference (MNLI) dataset contains three classes, *entailed*, *contradiction* and *neutral*. For these two datasets the problem would be a three-class classification problem. For legal domain entailment challenge, I formulated this problem as a binary classification task as in figure 1-2.

Given a training data $D = (x_i, y_i)$ for i = 1,..N

where

x_i = <texts, hypothesis>,

y_i = {0,1}

Figure 1-2 Problem formation of text entailment tasks in binary classification

This equation states that for given data *D*, a collection of samples and labels, classify *Y* as "1" if the two documents are entailed else "0". To solve the challenge of Natural Language Inference, I chose legal domain dataset. I participated in a Competition of Legal Information Extraction and Entailment (COLIEE 2018), on Canada case law dataset. Our task was to create a model that can provide decision for a new case that is entailed with the supporting noticed cases. The dataset consists of 181 training case, and 43 test cases. Each case contains a decision file which provides the decision for a base case, a summary file which summarizes the case and fact files which are human annotated facts to inform the reader about what events occur in the entire case. Along with these base case details, each case also provides a list of paragraph files as supporting noticed cases. The main idea behind COLIEE challenge was to derive a decision for a base case from the collection of supporting noticed cases, based on entailment. If a base case and a noticed case are entailed, then our classifier should classify those documents as 'entailed' and 'not-entailed' otherwise. Below is an example taken from COLIEE dataset to demonstrate the entailment relationship shared by two legal documents. 'Text' represents content of a base case and 'Hypothesis' represents an entailed supporting noticed case.

Text	Hypothesis
International Trust looks for some amendment to the Appraisal and Sale Order provision as to payment of moorage from the sale proceeds. Orders of the court, such as the present, are final, subject to some specific exceptions in the Federal Court Rules , whereby the court may correct its own errors and omissions, but such correction procedure is not appropriate here, for the order expresses the court's intention at the time the order was made. This brings International Trust up against the concept that, generally, once an order has been signed, it is final.	A perusal of the Rules of this Court reveals the general rule to the effect that an order is final, subject to an appeal, once it is signed by the Presiding Judge (rule 337(4)). Rule 1733 provides an exception to that general rule in cases where a matter arises or is discovered subsequent to the making of the order or on the ground of fraud. Counsel did not rely on rule 1733 nor was there any possible factual basis shown for the application of that rule. Rule 337(5) allows the Court to reconsider the terms of a judgment or order to ensure that it accords with the reasons or where there has been an accidental omission. Counsel did not rely, either, on rule 337(5). In any event, any application under rule 337(5) must be made to the Court "... as constituted at the time of the pronouncement ...". As noted supra this application to reopen was made to an entirely different panel of the Court.

Figure 1-3 Above example illustrates entailment between text and hypothesis for one of the base cases of COLIEE 2018, University of Alberta, Canada. Text column consists of decision and summary of a base case and the hypothesis is an entailed supporting paragraph for a given base case. (facts file is excluded in text while demonstrating as size of its text is large)

1.3 Contributions

Similarity and Relevance: Semantic similarity and relevance of two texts are used to establish entailment. The elements of information retrieval are included in this study by calculating relevance score of document pairs, to reason if relevance contributes towards establishing entailment. Here we try to analyze by only looking at the similarity, how much of inference is explained, and why similarity between two texts does not mean entailment. This experiment is designed using cosine similarity of documents and BM25 [Robertson et.al (2009)] relevance score to retrieve most relevant documents for further classification. We achieved our first higher scores than COLIEE winner using this technique.

Topological Data Analysis: Topological Data Analysis (TDA) is explained for inference. Motivation behind exploring geometry of text is to check how it can capture a structure in the data, which simple metrics and distance measure may not always provide. To establish entailment in specialized domain such as legal and dialogue systems, deep language understanding is necessary for achieving higher performance. With the help of topological structure, our model could understand more significant inter-document connectivity for establishing entailment. This technique achieved new state-of-the art results on COLIEE entailment task.

Graph Convolution network: I implemented a graph convolution network (Graph CNN) [Kipf & Wellings (2017)], a deep learning model, based on the graph that could capture information from its neighboring nodes for node classification. The choice of this network over traditional NN and LSTM was made by looking at the improved results of TDA. A combination of deep learning with graph yield the best results on establishing entailment on COLIEE task.

CHAPTER 2: BACKGROUNDS

In this chapter we introduce the basic terms and review related literature. Broadly it is divided into technical details of all the terms and techniques that are used for implementing this thesis.

2.1 TF-IDF: Term Frequency- Inverse Document Frequency

When mining plane text, it is necessary that we convert it into numbers which is in machine readable format. Representing plain text into machine readable numerical format is call "vectorization". There are a variety of methods available to create vectors for natural language. An algorithm or a deep network architecture can then learn from these vectors and interpret the results correctly. TF-IDF [Sammut & Webb (2011)] is one of the most useful and powerful techniques used when it comes to vectorization of text for machine learning algorithms to learn from. As the name suggests, it uses term frequency and inverse document frequency of words.

Term-frequency: It counts how many times a term occurs in a document. While computing term-frequency, all the terms in the corpus are treated equally important. It is recommended that we eliminate some of the stop-words such as, *an*, *a*, *the*, to ensure that the model does not give importance to such terms based on their high occurrence. Term frequency can be defined as:

$$TF(t) = \frac{number\ of\ times\ term\ t\ appears\ in\ a\ document}{total\ number\ of\ items\ in\ the\ document} \quad .$$

Intuitively, we can say that, terms that occur less frequently are more important and vice a versa. To compute this information, we use IDF, Inverse Document Frequency.

Inverse Document Frequency: We calculate the weights for each term based on their occurrence. This technique is slightly different than TF, because higher weights are assigned to those terms that are rare. Thus, it is a measure to calculate how important a term is. IDF can be expressed as:

$$IDF\ (t) = \log_e \frac{total\ number\ of\ documents}{number\ of\ documents\ with\ term\ t\ in\ it} \quad .$$

TF-IDF: It is a multiplicative term of TF and IDF defined above and can be expressed as:

$$TFIDF = TF * IDF \quad .$$

2.2 Cosine Similarity

In text analytics, to find the similarity between two documents, we calculate the distance between them. One of the most common approaches for achieving this is to calculate cosine similarity [Li & Han (2013)]. This metric gives a cosine angle between two texts and can help us determine how similar they are. If the cosine angle between the documents is closer to 1, they are similar. Cosine similarity is useful over other distance metrics considering it is unbiased towards the length of the documents. Let us assume an example where we calculate Euclidean distance between two documents. If two documents have relatively larger size, the number of common words between these two documents will tend to be higher. Using Euclidean distance which takes "common-words" approach into consideration can mislead us in determining that the documents are same because it can have same words but could be talking about completely different topics. Thus, by measuring the cosine angle between the two documents, we could insightfully examine how close those two documents really are and this gives more legitimate conclusion about their similarity. In mathematics, cosine similarity can be defined as,

$$\text{Cosine(A, B)} = \frac{A \cdot B}{\|A\| \cdot \|B\|} \quad .$$

Here A and B represent two documents. Before calculating cosine similarity, A and B should be converted into numerical format. One of the common approaches is to calculate cosine similarity on TFIDF matrix of document A and B. TFIDF can be calculated as explained in previous section.

Output of cosine similarity is $N * N$ matrix with diagonal elements set to 1 indicating similarity of each document with itself.

2.3 Relevance

Relevance plays a major role in information retrieval concept. When a user enters a query in a search engine, the engine retrieves a set of documents that are relevant to the query entered by the user. Relevant documents are returned by calculating the occurrence of query term in the corresponding document terms. Considering the presence of query terms, all documents with query terms in it, would be ranked according to their relevance with the query. High scored documents will be returned at the top followed by lower ones. Although the concept of search engine implementation is out of the scope of this work, we employ one of the search engine methods.

BM25: BM25 stands for Best Match scores. This is one of the most powerful algorithms that can rank the documents according to their relevance to the query. Given a query Q, containing keywords [q1, q2 , q3 , ….qn] the BM25 score of document D is:

$$score(D,Q) = \sum_{i=1}^{n} IDF(qi).\frac{f(q_i,D)(k1+1)}{f(q_i,D)+k1\,(1-b+b.\frac{|D|}{avgdl})} \quad .$$

Where; $f(q_i,D)$ is $q_i's$ term frequency in document D, $|D|$ is the length of the document D in words, $avgdl$ is the average document length in the text collection from which documents are drawn. k1 and b are free parameters, taken in absence of an advanced optimization, as k1 ∈ [1.2, 2.0] and b is 0.75. $IDF(q_i)$ is the IDF weight of query term q_i and it is computed as:

$$IDF(q_i) = \log\frac{N-n(q_i)+0.5}{n(q_i)+0.5} \quad .$$

Where N is total number of documents and $n(q_i)$ is number of documents containing q_i query term.

BM25 represents the best probabilistic relevance. This algorithm is also unbiased towards document's length. For example, if we consider calculating relevance based on query terms occurrence only, lengthy documents with more matching terms could be retrieved as relevant document where they are not the best to answer the user's query. BM25 overcomes this limitation.

2.4 Topological Data Analysis

Topology is a classic branch of mathematics that deals with shape invariants such as the presence and numbers of holes. More recently topological data analysis (TDA) was introduced as a branch of computational mathematics and data science, predicated on the observation that data points have implicit shapes [Edelsbrunner and Harer (2010)]. Both topology and TDA can be viewed as an abstraction mechanism, where we replace the original shape or cloud of data points by some numbers representing their mathematical properties, using a formal machinery derived from algebraic topology.

Figure 2-1 [Huang et al. (2018)] conveys these ideas: it shows a cloud of data points, and its subsequent approximation by balls of increased radii. The overlaps produce a change in shape which can be measured using the H0 and H1 lines: The number of H0 lines intersecting the vertical bar at ε is the number of connected components of when the points are extended with balls of that radius. Therefore, as increases, the number of components decreases. In this process the exact values of the data points are ignored, but the shape information is preserved – that is, two clouds of similar shapes but different values will have similar persistence diagrams. The H1 lines show the birth and death of holes at given values of ε . The top line shows a hole persisting from 1.2 to 3.3 (approximately). Jointly, H0 and H1 compress information about the shape of the point cloud. This diagram deals only with planar structures, but persistence works in higher dimensions as well, in principle allowing machines to "see" shapes in dimensions higher than 3, a task difficult for humans. "Persistence" refers to the fact that the number of components and holes remains stable at

some intervals, and we record this fact as numerical features; “homology” means similarity (of shape).

In NLP, the points are in a high dimensional space and represent vectors of TFIDF or other features derived from text. The method works the same, but please note that figure 2-1 only illustrates how TDA progresses from points to shapes. At this point, we do not know, and we see it as a major open problem what aspects of natural language semantics, whether for entailment or classification, are captured by topological features.

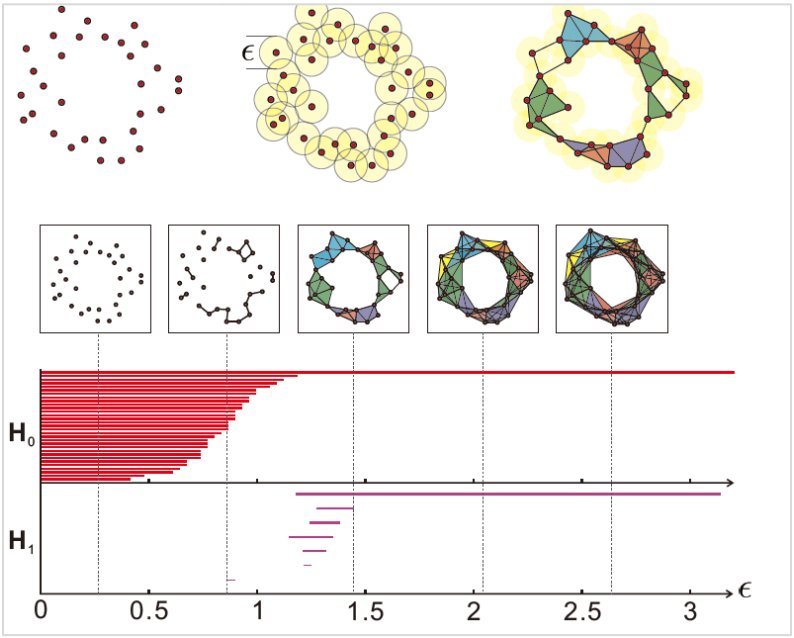

Figure 2-1 Persistence homology is a data analysis tool. Intuitively, as we start expanding the data points into balls of increased radii, planar figures emerge and change. The intervals in H0 and H1 capture relevant features of this process, namely the number of connected components, and the number of holes at different resolutions. The method abstracts distance information about the feature vectors of original data. It is an open problem how exactly these new numerical features help entailment. [Huang et al. (2018)]

To finish this introduction, we mention an equivalent representation, called persistence diagram, demonstrated in figure 2-2. Persistent diagram represents birth and death as two-dimensional coordinates. To repeat, the representation method is general, and it generates numbers we can use

as machine learning features. However, finding the corresponding natural language mechanisms responsible for the improvements in accuracy of classification or entailment is an open problem. Below is an example of persistent diagram.

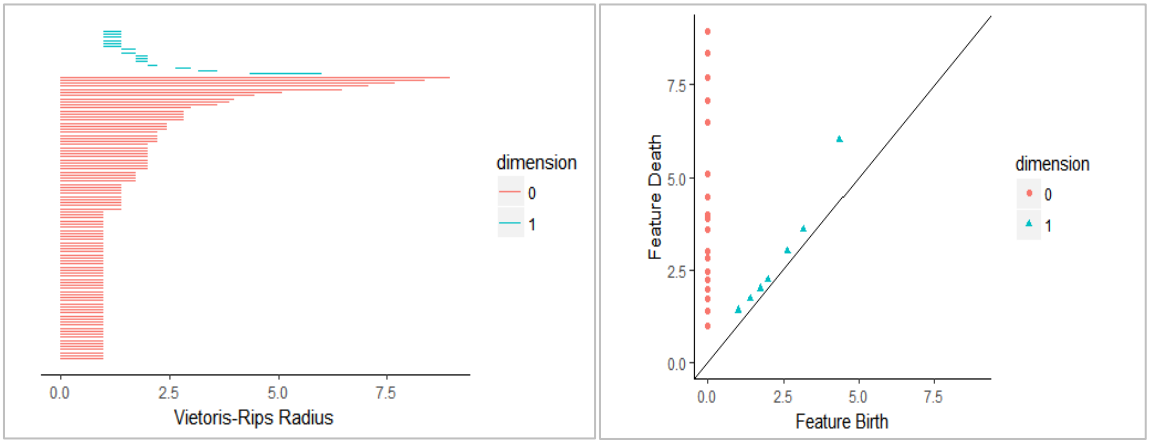

Figure 2-2: Persistent diagram constructed on one of the COLIEE cases. Left Panel: Barcode structure of persistent homology capturing multiple cycles. Note the differences in radius of one long cycle and the others. Right panel: Persistent diagram representing the cycles from the left panel. Note the dot further from the diagonal corresponding to the long cycle.

2.4.1 Topological data for classification

Applications of TDA to text started with discourse [Zhu (2013)] used nursery rhymes to illustrate properties of homological persistence (e.g. that it is not simply measuring repetitions), and showed that children, adolescent and adult writing styles can be differentiated using TDA. [Doshi and Zadrozny (2018)] used Zhu's tools and methods to show that topological features can improve the accuracy of classification (movie plots). They also discuss the paucity of applications of TDA to text, and the fact that not all these applications show improvements over the state of the art: in particular this was the case for sentiment analysis and clustering [Michel et al. (2017)]. [Temˇcinas (2018)] argues for applicability of persistent homology to lexical analysis using word embeddings, and for discovery of homonyms such as 'bank', thus potentially for word sense disambiguation.

For discourse analysis, broadly speaking, [Guan et al. (2016)] TDA can help with extraction of multiword expressions and in summarization. Also, [Horak et al. (2009)] applied TDA to a network of emails, without going into their text. TDA for text data is an emerging area of research, with a potential to be of value for computational linguistics.

2.5 Graph-Convolution Network

Graph CNN is relatively new yet very powerful technique. [Kipf & Wellings (2017)] introduced it by experimenting it on various graph suitable datasets such as Quora and Citeseer. Graph CNN is a neural network design that can learn the pattern from the designed graph. Graph CNN can provide useful feature representation of nodes in network. The input to the graph will be an adjacency matrix and feature matrix, edges with their weights and labels. This architecture could be used to establish node classification task as well as edge classification task. The idea behind creating a graph is to create a structure that establishes the connectivity between words and documents present in the corpus.

2.5.1 Graph CNN for Text Classification

[Yao et al., (2018)] constructed a single text graph which is based on word-cooccurrence and document word relation. This structure can be used to learn graph convolution for text classification. Graph convolution networks can be useful for node classification as well as edge classification. In this work, we will use them for node classification. The Graph CNN also learns predictive embeddings.

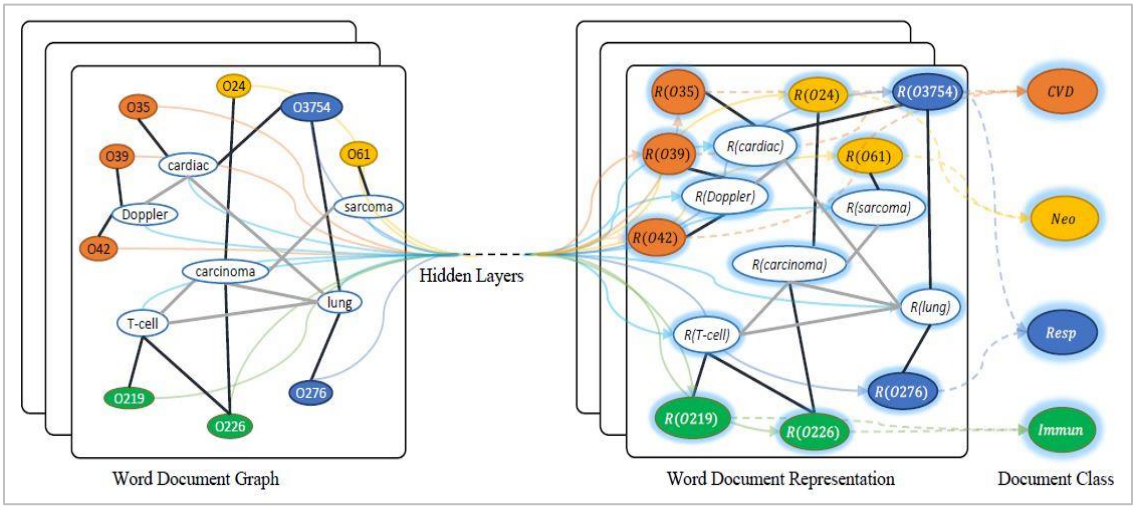

Figure 2-3 Schematic of Text GCN. Example from Ohsumed corpus [Hersh et al., (1994)]. Nodes begin with "O" are document nodes, others are word nodes. Black bold edges are document-word edges and gray thin edges are word-word edges. R(x) means the representation (embedding) of x. Different colors mean different document classes [Yao et al., (2018)]

In graph, nodes are words and documents available in corpus. The size of the nodes is total number of words that is our vocabulary size plus total number of documents that is train as well as test documents. Each document in the network is a combination of decision, summary and fact file with corresponding paragraph from the list of noticed cases. If the base case (decision, summary and fact file) is entailed with the corresponding noticed case, then that node will be classified as 'entailed' node and 'not-entailed' otherwise. The edge is a connectivity between all the nodes. Considering we are using word co-occurrence; we would be using pointwise mutual information (PMI) as a weight of the edge between two words. For establishing connectivity between word-document nodes we will use weight of the edge as TF-IDF score which indicate the inverse document frequency of a word in given document. Pointwise Mutual Information (PMI) for the word pair *i, j* can be generated by the formula blow:

$$PMI(i,j) = \log \frac{p(i,j)}{p(i).p(j)}$$

$$p(i,j) = \frac{\#W(i,j)}{\#W}$$

$$p(i) = \frac{\#W(i)}{\#W}$$

In the above equation, #W (i) is the number of sliding windows in corpus that contain the word *i* $\#\ W\ (i,j)$ is the number of sliding windows that contain both the words *i* and *j*. $\#W$ represents the total number of sliding windows. If the corresponding PMI is positive, then that indicates high semantic correlation of words in a corpus whereas negative PMI indicates little or no correlation. Edges will be established only for positive PMI between two words.

Input to the graph will be an adjacency matrix and features. When it comes to convolution nets to understand the connectivity of graph, we need to define the connectivity of two or more notes as adjacency matrix. It is an $N * N$ matrix which is passed as an input with feature matrix and labels to the architecture. For designing an adjacency matrix, we need to index our document nodes. The vocab nodes could be easily identified along with the connected IDF scores in between vocab and document. While creating the adjacency matrix, we create a list of document names and fetch their indexes in memory. While defining the connectivity, we can define the connectivity with respect to their indexes. Below is an example of smaller adjacency matrix:

If there is an adjacency matrix for 2 documents, then, Adj = $\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ Here the first row and first column represent first node which is a self-connected node. Similarly, if two nodes are connected then the corresponding entry in adjacency matrix will be 1 else 0. In Graph CNN the edges are PMI

and TFIDF. Calculating the sum of all the edges forms a degree matrix $D = \sum_j A_{ij}$. This is used to calculate the normalized adjacency matrix A = $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$.

Feature matrix as an identity matrix denoted as $X = I$ of features because we want to represent our text in one-hot encoded format. We allow Graph CNN to learn the best representation for our text and create predictive embeddings to represent the same. Along with adjacency matrix and feature matrix, we feed train data and train label and all connected nodes and corresponding labels to the graph. While we train the network, we mask the labels of the test data so that the graphs cannot see them. Labels for vocabulary will be equal to zero. In this architecture the main idea in theory is, the graph learns very well from even the small training data and yet can generalize the test data to give accurate predictions. The main difference of Graph CNN and traditional convolution neural architecture is, Graph GCN does not have traditional Maxpooling and flatten layer along with Conv1D or 2D, instead it uses the Convolutional Graph layer to convolve over the graph represented using adjacency matrix. By convolving over matrix, it learns how the nodes are connected with each other and establishes the node classification.

If the graph has two layers, then it shares the details of two of its neighborhood nodes and convolves further with next window. If we want to increase the number of neighborhood nodes through which information is being propagated, then we must increase the hidden layers of the graph.

2.6 Random Forest

As Random forest [Kerry Raymond (1989)] is a collection of decision trees. The basic idea behind trees is that given a set of observations and predictors, it creates a subsample of features and observations to decide the outcome of the combination. If we wish to predict a continuous value, we use regression trees. If our aim is to predict categorical values, then we use classification trees. Trees are very powerful choice especially when the data is not normally distributed.

2.6.1 Decision Trees

The primary motive of decision tree is to create a training model which can use to predict class or value of a target variable. It learns decision rules inferred from the data. The tree considers the entire dataset as one and tries to split the data into root node. A root node is the node that contains maximum information to decide while moving on. The decision trees can be programmed to split the data further using either most purity-based split for the end nodes or maximum information gain to obtain the pure node at the end of the splits.

Information Gain: By using the Information Gain measure, a tree obtains the node with maximum information. Information Gain is calculated based on "entropy" which simply measures how random given data is. When we are using trees for entailment task, we will consider establishing a rule based on binary class 0 and 1. If majority nodes contain class 0, there is less randomness, which will have lower gain. If some nodes are 1 and rest are 0 for a given class, the entropy will be high. Information gain can be expressed as

$$H(X) = \prod [I(X)] = -\sum p(X) \log p(X) .$$

Where $H(X)$ is an entropy, explains randomness. Expected value of information gain $I(X)$ can be calculated as sum of product of all probabilities multiplied by their log probability. Here the negative sign is maintained to ensure that the outcome is positive as information cannot be negative. This process will be iterative until we get pure nodes. The common issues with decision trees are that they suffer from overfitting problem by learning the data too well. This leads to failure of generalizing the test data. If there is only one tree, it won't learn the best representation of the data to be able to classify correctly. This is the reason why grouping multiple decision trees and allowing them to aggregate the results gives better performance.

2.6.2 Random Forest

Random Forest, as a collection of decision trees, aggregates the classification performed by every decision tree and decides final class for given training sample. It is an example of ensemble model. When we combine multiple trees together for modeling random forest, it can perform more robust predictions without suffering from overfitting problem. More the number of trees better are the results in Random forest however this may vary for certain datasets.

Figure 2-4 is an example of the random forest model where grouped decision trees of multiple instances are used. It is to aggregate the predictions of each tree and then in second pass to provide final answer that contains majority voting of trees indicating sample belongs to a class. For splitting the nodes and calculating the end node it uses information gain, same as decision trees.

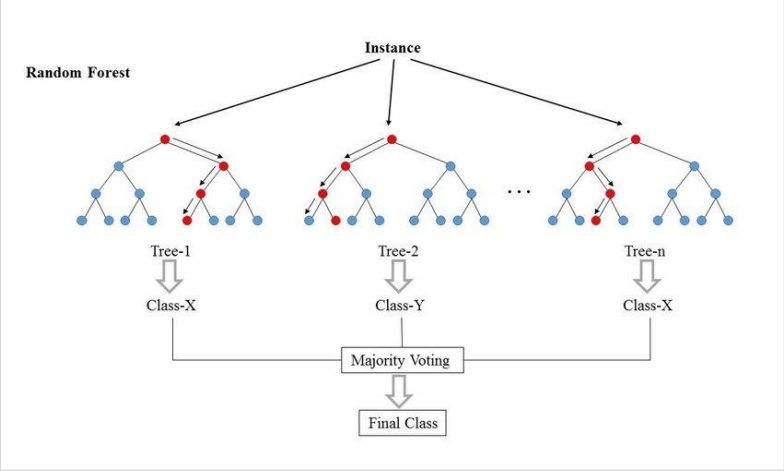

Figure 2-4: Tree structure explains the mechanism of Random Forest. It shows how the nodes are getting split to infer the class based on different feature selection. By considering majority of the classification decision by several trees, it will output the actual class to which given data points belong to. [article by Niraj Kumar, (2016)]

2.7 Class Imbalance

In real world, the data does not come in normal distribution format. There is always a possibility that total number of positive classes are greater than total number of negative class or vice-a-versa.

Some of the well-known examples are fraud detection where in a given set of records approximately only 1% of records are fraudulent and rest are genuine transactions. In medical tests, where small population is suffering from disease and hence it is difficult to correctly identify the actual affected patients.

2.7.1 Resampling techniques

The most commonly used techniques for balancing the classes are oversampling, under-sampling and hybrid. In oversampling, the method increases the number of minority class samples where as in under-sampling, the method decreases the number of majority class samples. Hybrid resampling technique is a mix of under and oversampling.

NearMiss under-sampling: This algorithm calculates all the distances of majority class and the minority class. In first pass, the distances between the majority classes are calculated followed by minority class. We define the value for k number of instances of the majority class which have smallest distance with minority class. Depending upon the value of k, the number of samples for majority class will be selected. For n number of instances in minority class, this method results in $n * k$ instances of the majority class. There are three versions available for this algorithm, NearMiss 1, NearMiss 2 and 3. NearMiss 1 selects 3 majority classes whose average distance is closest to the 3 instances of minority classes. NearMiss 2 selects 3 farthest samples of minority classes and NearMiss 3 selects k number of closest majority class for each sample of minority class.

CHAPTER 3: PROPOSED METHODS

Selection of approach to establish entailment highly depends on the dataset we use and the task for which we want to establish entailment. In this section we will discuss my approach for solving entailment task on COLIEE data. Figure 3-1 illustrates the pipeline of Similarity with Relevance and Topological model implementation. I first preprocessed the data by normalizing the text. In feature engineering, I took care of class imbalance by selecting highly similar and relevant documents for every base case. I under-sampled the data with NearMiss version 3 algorithm. For vectorization, I used TFIDF vectors and used the data for classification to establish entailment. In the topological data analysis model, I concatenated topological features with TDIDF vectors, keeping data preparation the same. Details of pipeline are mentioned in section 3-1 and 3-2.

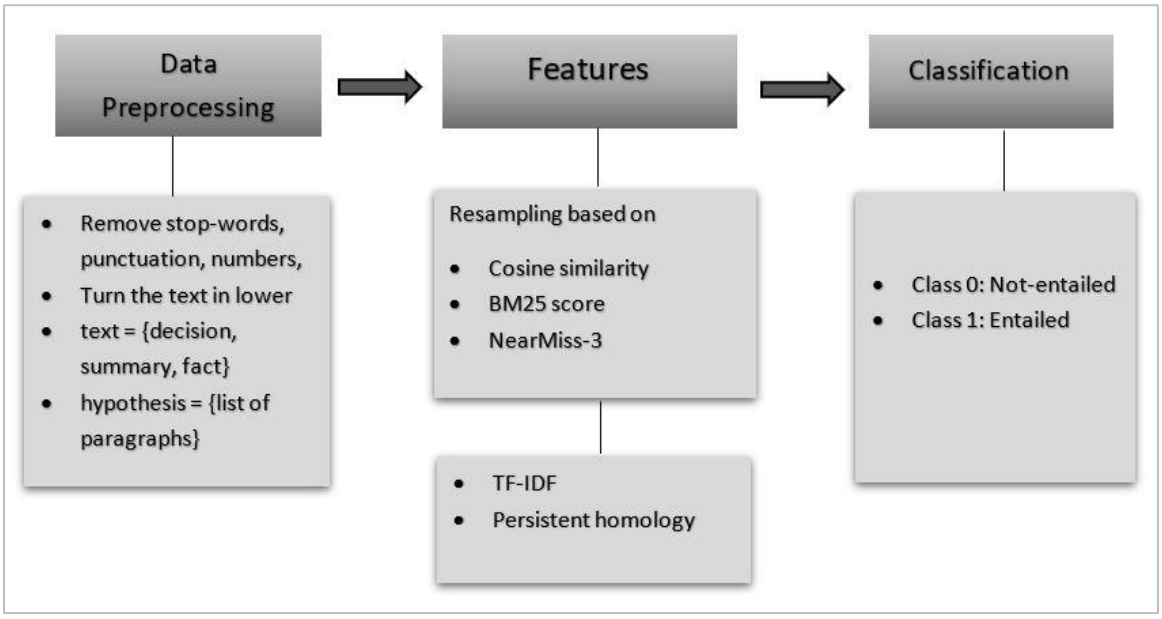

Figure 3-1: Diagram represents pipeline used for establishing entailment. A simple flow was to pre-process the data, prune highly similar and relevant paragraphs and resample further using NearMiss-3, then in the second pass, use our models, similarity and relevance-based entailment classification, TDA for entailment and Graph CNN for entailment.

Class Imbalance: At the beginning of the implementation of every model, I had to take care of class imbalance problem. Considering the data has 97% negative class, I observed that the neural

network architecture as well as Logistic Regression classifier was giving 97% accuracy. This gave a false impression of great start as the model was predicting all classes as class “0” by simply discarding all class “1” as noise. After trying to assign higher penalty to CNN and LSTM’s cost function also there was no great improvement in prediction results. This gave me a clear indication of how difficult this task is! I also tried tuning “class_weight” hyperparameter for machine learning classifier and using Boosting classifiers to reduce misclassification but nothing helped. This behavior of model required special attention to tackle class imbalance without losing any information from data while classifying it. In this thesis, I implemented two classification models for establishing entailment. In this section we will discuss about model architecture and intuition behind the same.

Considering the major challenge was class imbalance ratio of 3% positive to 97% negative, I first focused more on reducing this severity. For given cases, we had at least 10 and at most 250 noticed cases. Out of which at least 1 and at most 4 were entailed cases. For our first approach, I calculated cosine similarity between a base case and each paragraph file. BM25 scores is used to rank paragraph files for their relevance with the base case. This helped retaining the top 10% of highly similar and relevant paragraphs with the base case for further classification. Our hypothesis was, combining these features to re-sample the data should maximize the probability of establishing entailment without any information loss. I further used NearMiss3 algorithm for under sampling the majority class data. The class imbalanced was tackled in train dataset however it was the same for test dataset. My goal was to train the model with no information loss while handling imbalance and improve its generalization in prediction of test data.

3.1 Similarity and Relevance

To begin with the very first model, text should be converted into TFIDF vector representation for classification task. Considering this problem is based on legal domain, one must obtain sufficient

amount of legal data to create embeddings else, use traditional vectorization. In this experiment considering there is a paucity of legal data, computation of legal embedding was difficult. For text to have the best representation and to avoid zero vectors for most of the vocabulary, TDIDF representation works the best. This experiment can be performed using Random Forest classifier for binary classification. Knowing the robust nature of Random Forest, it performs best over other classification algorithms. This choice of classifier also helps in reducing false positive predictions when the data is not uniformly distributed. Data preprocessing step begins with removing stop words from the corpus such as *a, an, the, of* using NLTK [Bird et al.,(2009)]. After removing the stop words, punctuations such as "?, ,, . !" as well as special characters and numbers such as "$#@%&" and "1[12][2]" etc. should be removed. For normalizing text, convert the text in lower case so that words like "Law" and "law" should be treated equally by our classifier. After completing the text normalization, this problem becomes a supervised machine learning problem. Here, one must define premise as a combination of decision, summary and fact files for each case and hypothesis as corresponding list of noticed cases for every base case. All the corresponding list of paragraphs to be labeled as '0' if it is a non-entailed paragraph and '1' if entailed. Label information was provided in an XML file.

The values of cosine similarity ranges between 0 to 1. If two documents are very close to each other, then the corresponding similarity will be 1 else towards 0. Using the IDF element of TF-IDF and Python's Gensim package [Rehurek & Sojka, (2010)] BM_25 scores for every paragraph, relevance can be calculated. This element is very useful in choosing the most relevant paragraphs for a given base case based on how relevant they are with the base case. Element of relevance can explain exactly which documents to pick for further entailment task. To determine the relevance factor, IDF score of query term in corresponding document should. If the query term appears in the document with relative proximity, that document will be considered as relevant document for our query. In this setting our query is a premise. Using the values of cosine similarity and BM_25,

those paragraphs with high relevance score and high similarity value were shortlisted for further classification task. The reason behind not simply concluding this experiment basis similarity is because documents that are highly similar may not be the entailed one. Thus, the choice was made for top 10% paragraphs depending upon the number of supporting paragraphs in each case. After shortlisting the paragraphs, imbalanced can be slightly reduced but not completely tackled. For implementing the Random Forest classifier, I used simple sklearn implementation of Random forest.

3.2 Topological Data Analysis for Text Entailment

Unconventional technique called Topological Data Analysis is employed in this model to examine document entailment relationship. Topology can explain a structure which could create stronger signals to capture entailment. From the previous method, entailment cannot be entirely explained only by establishing similarity. By measuring a distance between the two documents, one cannot necessarily infer a meaning of one text from another. In Information Retrieval, if a document is relevant to a given query, it does not necessarily mean that the meaning of a query can be completely inferred from the retrieved document. In fact, this creates a need for new effective text representation for entailment.

In this method, the initial part of data preprocessing and class imbalance remains the same. After normalizing the text, model should select most similar and relevant paragraphs by calculating cosine similarity and BM25 scores. For computation of homology I used decision, summary and fact file along with highly relevant and similar corresponding paragraph files for each case at a time. For every single case, I then created a TFIDF vector of all the files available in the case. Use of Ripser, a C++ library [Bauer, U. (2016)] to compute persistent homology, for establishing topological structure of documents is the efficient method to compute homology. This package forms simplical complexes between files available in each case. The output of topological features

is a birth and death radius of each topological feature. This formation of simplical complex is controlled by defining betti numbers values. For topological features, the assumption is, if text is entailed with hypothesis then the corresponding values of birth, death radius could provide stronger signals to the classifier. After calculating homology, I concatenated persistent homology features with TFIDF to create a feature vector comprised of the same. Here I mapped the homology features back to each case and concatenated the two feature vectors, homology and TFIDF as an input for binary classification task. I used Random Forest classifier to establish entailment. The motivation behind using Random Forest for this experiment and not any other classifier is that, I wanted to examine the contribution of topological features under the exact same settings. Instead of changing the model that could perform better, the idea was to see what homology can do to the relationship establishment and when it comes to giving stronger signals to the model to learn from.

3.3 Graph CNN for Text Entailment

To incorporate elements of deep learning in this research we adopt Graph Convolution Network [Kipf & Wellings (2017)] as our choice of third model. In this network I have designed a graph mainly on COLIEE dataset and validated the approach by implementing it on SNLI and MNLI dataset. As discussed earlier, COLIEE data has a base case and corresponding list of paragraphs with each base case. These paragraphs are the noticed cases from which our model should identify an entailed case which can explain the decision of a base case. We formulated this problem as a supervised machine learning problem as discussed in previous chapters. Our work is primarily inspired by Yao et al., (2018) who employed this technique for node classification task. This research primarily focuses on establishing a graph on unstructured documents and vocabulary. By showing the word-document and word co-occurrence they proved that even though the training data is small in size, the test data can be well generalized by graphs. In this technique they have achieved a new state-of-the-art accuracy of 93% on 20ng dataset. Although there is verity of graphs

available, here the choice was to go with [Yao et al. (2018)] graph CNN to examine if given a set of document and vocab, can graphs capture inference? As [Yao et al., (2018)] also used it for classification of document corpus such as 20ng, it was intuitive choice to check if it can learn a harder problem of classifying documents as entailed documents.

Construction of Graph: I have used Graph CNN for node classification problem. In COLIEE dataset, for graph construction, I prepared the data for node classification. Post this step, combined base case with corresponding supporting case, one at a time, to determine if the node is an entailed node or a not-entailed node. I began by reading the text corpus and followed the same steps of preprocessing. For labeling as an id of documents, I used documents with their file names given in corpus. The list of all the names of base case document names and corresponding noticed cases document names was created. For example, for case 1 the corresponding document node will have a label as “001.txt##01.txt”, “001.txt###02.txt” and so on. After storing the names of all the files in a variable, I then created a list of an actual content of all the files for base case and supporting noticed cases. Post this, the content I focused on was creating the ID list for all the document. This step is necessary for mapping our data in node connectivity. Initially I assigned train and test Ids as the name of the files and maintained all the ids together in a variable. In graph, we always use the entire data by masking the labels of test set and thus, this step is crucial.

Once the IDs are computed we can move on to creating the vocabulary from corpus. As our graph contains word co-occurrence, this begins from creating a vocabulary. By extracting the unique words from given corpus, we can obtain vocab and use it as word node. After extracting all unique words from corpus, I created word-document list. This step was needed as our graph consists of word-word and word-document connectivity. Here the correct mapping of vocab is crucial. When we connect words with documents through an edge, we want to make sure that the corresponding node has correct vocabulary in connection for classifying it accurately. The graph always shares

information with its neighborhood nodes and thus correct mapping of vocab is important. Once the nodes were established, I created the edges between word-word and word-document. For two words to be connected, I calculated pointwise mutual information between two words and connected them basis the value of PMI. For word-document edge, if IDF value of word is higher in respective document then that word and document was connected by an edge.

After establishing nodes and edges I created train and test set of documents with their labels as sparse csr matrix to feed as an input to the graph. At this stage of computation, I created train data with labels, test data with labels and all nodes connected as an input. Here since this is a node classification problem, one important distinction is every node has a label associated with it. But considering words cannot be labeled as entailed or not entailed, we need to label it as zero. While creating fully connected graph using train, test documents and entire vocab nodes, I have stacked all the train and test labels along with zero labels for vocab nodes to ensure compatibility of shape between number of nodes and number of corresponding labels.

For COLIEE, I created two-layer convolution graph CNN. This architecture is slightly different than [Yao et al., (2018) and Kipf and Wellings (2017)]. The reason behind tuning separate parameters is for neural network to learn our complex problem better. Tuning of hyperparameters and their optimum values are discussed in chapter 4. For filter, I have used the default choice of Kipf and Wellings' Laplacian filter. This filter is designed by creating symmetrical normalize adjacency matrix. This matrix can be calculated by taking a dot product of adjacency matrix with its transpose. The Laplacian can be calculated by subtracting normalized symmetric adjacency matrix from actual obtained adjacency matrix. The author then obtained eigen values and eigen vector from Laplacian.

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \quad .$$

Where I_N is Identity matrix.

Using Laplacian polynomial of K^{th} order one can employ convolution in this architecture.

CHAPTER 4: EXPERIMENTAL SETUP

This section consists of details about COLIEE, SNLI and MNLI dataset. In section 4.2, I shall discuss multiple data preprocessing steps followed for three datasets. As I have implemented three different models, which are discussed in section 4.3. In 4.4 we will discuss about performance metrics Precision, Recall and F-measure along with brief reasoning behind selecting these methods over other available methods. All in all, this section focuses more on practical implementation of the project.

4.1 Datasets

4.1.1 COLIEE 2018

To execute this thesis project, we mainly worked on a **C**ompetition on **L**egal **I**nformation **E**xtraction and **E**ntailment (COLIEE 2018), Task 2. This task consists of establishing entailment between base case and its corresponding supporting case to derive a decision of a case. For training, there were 181 base cases provided which were drawn from an existing collection of Federal Court of Canada law cases. Every case consists of a decision file, summary file, facts file and a list of paragraph files. The training data also consists of labels in XML format for entailed paragraphs. Our task was to identify paragraphs from this list, that entails with the decision of a base case. In 181 base cases, the number of paragraph files were 8794 out of which 239 were positively entailed and the rest were not entailed. This led us to a class imbalanced ratio of 2.71% positive class and 97.29% negative class.

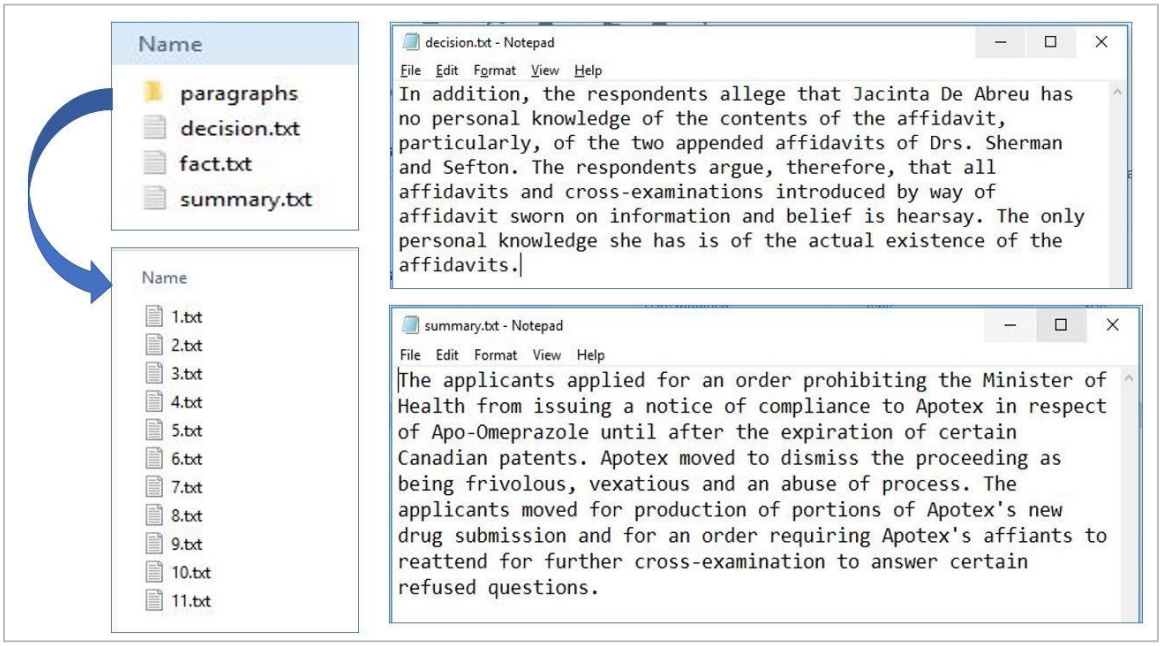

Figure 4-1: Each case folder includes decision file, summary file and fact file with paragraph folders. Decision file is an actual query i.e. a decision of a base case, summary file consists of a summary of a base case and facts file includes all the human annotated facts about the base case.

To solve an entailment task, given a decision of a base case, along with its summary and facts, the system should be able to establish the relation of entailment with an associated noticed case, given as a list of paragraphs. More formally, given a base case b, and its decision d, and another case r represented by its paragraphs P = {p_1, p_2, p_3,….. p_n} and we need to find the set E = {p_1,p_2,….,p_m | $p_i \in$ Pg,) where entails(p_i; d) denotes a relationship which is true when $p_i \in$ P entails the decision d [Rabelo et al. (2018), Kim et al. (2016), Adebayo et al. (2016)]. We then formulated the problem as a binary classification problem for establishing corresponding paragraphs as entailed or not entailed with a base case. Mathematically, given a training data $D = (Xi, Yi)$ for i = 1,…N,

where x_i = <texts; hypothesis>; and $Yi = \{0,1\}$

Why is it difficult?

For legal domain dataset, we require an understanding of law to analyze it. A traditional approach such as training neural network, or the more intuitive semantic similarity approach did not work very well on this dataset. Reason being, pre-trained word embedding such as GloVe and word2vec may not contain enough legal terms for neural networks to learn. Also, a corpus is not too large to create our own pre-trained word embeddings in the legal domain. Another challenge was data distribution. Using common re-sampling techniques for classification task along with TFIDF only, generalized its predictions as negative class by treating positive class as noise and gave false high accuracy.

4.1.2 SNLI

In SNLI which is Stanford Natural Language Inference dataset, which consists of 570k human annotated sentence pairs, in English language. These pairs are manually labeled for 3 classes, Entailment, Contradiction and Neutral. If we can obtain a meaning of one sentence from another then the relation between the two would be ‘Entailed’. If two sentences are expressed in opposite direction, then they will be labelled as ‘Contradiction’ and if two sentences have no relationship with each other then they have been labeled as ‘Neutral’. This is a 3-way classification problem where given a text *t* and hypothesis *h*, the task is to predict whether *h* is entailed by *t*, *h* is contradicted to *t*, or whether the relation between *t* and *h* is neutral.

Below are some of the examples of NLI task:

Table 4-1: Example of NLI task consists of sentence 1 which serves as a premise, sentence 2 which serves as a hypothesis and relation indicates how premise and hypothesis are related to each other. If meaning of hypothesis can be inferred from premise, then they are entailed with each other. If premise and hypothesis are indicating opposite meanings, then they are contradictory statements and if they do not share any common though then they tend to have neutral relationship

Premise	Hypothesis	Relation
Conceptually cream skimming has two basic dimensions - product and geography.	Product and geography are what make cream skimming work.	Neutral
One of our number will carry out your instructions minutely.	A member of my team will execute your orders with immense precision.	Entailment
Vrenna and I both fought him and he nearly took us.	Neither Vrenna nor myself have ever fought him.	Contradiction

.

4.1.3 MNLI

Similar to SNLI, NYU- Multi Genre Natural Language Inference is a more generic dataset with 433k samples. They also belong to 3 classes, Entailed, Neutral and Contradiction. It is more challenging than the SNLI when it comes to achieving good performance on MNLI considering the length of the sentences is relatively larger than SNLI. Also, these are extracted from real world communications and have genres for the text. This corpus is in English language with balanced classes. Below is an example for each class drawn from MNLI dataset.

Table 4-2: Examples drawn from MNLI dataset. The first sample shows that it is extracted from telephone conversation. Since Premise can be inferred from hypothesis, the relationship between these two sentences is entailed. Similarly, in the second and third example, since they two are opposite to each other and share no common fact respectively, they are labeled as contradiction and neutral.

Genre	Premise	Hypothesis	Relation
Telephone	well you see that on television also	You can see that on television, as well.	Entailment
Travel	Fun for adults and children.	Fun for only children.	contradiction
Government	Conceptually cream skimming has two basic dimensions - product and geography.	Product and geography are what make cream skimming work.	Neutral

4.2 Data Preprocessing

For COLIEE dataset we preprocessed the data simply by using NLTK package of python. This step involves normalizing the text, turning it into lower case, removing English stop-words and extracting the roots of the text. For vectorization we used TFIDF scores of words on corpus level. After normalizing the text, extracted labels from corresponding XML file. We then formed it as a supervised machine learning problem by framing [decision, summary, fact] as premise and noticed cases as hypothesis.

In SNLI and MNLI, we dropped all the columns except for “gold_label”, “sentence1” and “sentece2”. The value of target variable “gold_label” was “contradiction”, “Neutral” and “Entailed”. We converted them into categorical values. We assigned 0 to class contradiction, 1 to class entail and 2 to class neutral. We then transformed these values into one-hot vector for target variable. Unlike COLIEE, in SNLI and MNLI dataset, words like is, not, isn’t, etc played important role of establishing polarity. Due to this, we did not remove stopwords while creating tf-idf vectors.

Polarities of sentences are very important while establishing entailment. One can say that positive sentence cannot be inferred from a negative one.

4.3 Model setting

In this section we will discuss all three models. My primary goal was to solve COLIEE and check model performance on SNLI and MNLI dataset. Considering the nature of the two problems is different, certainly I have made some adjustments in data processing as discussed in previous section.

4.3.1 Similarity, Relevance and Random Forest

While defining a problem as supervised machine learning model, I represented all data points as a set of elements of type "[text, hypothesis], Label". In our entailment problem, "text" is a combination of decision file, summary file, fact file and "hypothesis" is a list of paragraphs for a case. For cleaning the text data, we simply removed punctuation, stop-words followed by converting the text to lower case and stemming it all using python NLTK package.

For SNLI and MNLI we followed the same procedure and the only difference was this is a multi-class classification problem. We had to define our output Y as one hot vector of three classes. The output of the random forest classifier was 2^3 size output vector for each premise and hypothesis.

4.3.2 Topological Data Analysis for Entailment

In our second model, keeping the base as first model we added topological feature to the feature vector of classifier to examine if topology can provide more information to the machine learning algorithm.

Computation of Topology for COLIEE data: This task was implemented on each case level. Given a base case and a supporting noticed case, we first computed TFIDF vectors of each file. We then computed cosine similarity scores for a base case and every corresponding paragraph per case at a time. After computing the distance, we used it as an input to "Ripser" for computing persistent

homology for each case. Considering the input to homology was a case at a time, the algorithm could traverse between the set of files available in each case and compute topological cycles, if any, between base case and supporting case. This cycle could be expressed as an output of persistent homology. The output of algorithm consists of birth and death radius on dimension 0 as well as for dimension 1. We then mapped these features back to each case and fed them to the classifier.

Computation of Topology for SNLI and MNLI data: In SNLI and MNLI we primarily have 3 columns in dataset, sentence 1 which is *premise*, sentence 2 which is *hypothesis* and gold_label which is a target for each sentence pair. Similar to the above method we computed TFIDF vector and cosine similarity for sentence 1 and sentence 2 to prepare the data for topological feature extraction. Ran Ripser in iterative manner on one sentence pair at a time. Here I observed that because the sentence length was smaller in size, it could not obtain many cycles of dimension 1 in data. Although we obtained dimension 0 birth and death radius, they were powerful for the machine learning algorithm to understand a structure of a data.

For all three datasets, we computed Random Forest classifier using sklearn [Buitinck et al., 2013] in python 3. We tuned the hyperparameter of a classifier as it is, to examine and validate the feature selection and how it works given different datasets. For Random Forest classifier we only tuned n_*estimator* parameter and *criterion* parameter. We chose value for n_estimator as 200 and "entropy" criterion. The reason behind choosing entropy over Gini is because we wanted to give more importance to information gain. While it performed very well on COLIEE, it could not generalize the NLI task very well. Here my observation is that, it is certainly not the feature but the classifier which needs advanced selection for correctly establishing relations for NLI task. Research shows that LSTM networks are delivering promising results and I would surely use word embeddings with topological features for LSTM while improving my results in future work.

4.3.4 Graph CNN

In this section we will discuss about how I have implemented graph CNN for COLIEE data followed by SNLI and MNLI dataset. The core implementation of methods and model is inspired by Kipf and Wellings (2017) along with Yao et al., (2018). There were 28298 nodes available in the graph for COLIEE and it took 3 hours for the model to converge. For training the network I used 200 nodes in hidden layer, 0.7 dropout rate, learning rate 0.02, weight decay 0 and 600 epochs without early stopping. Here the best result obtained was 45% F-measure, precision and recall. Here I improved my result by 2% from the previous method. Although the increment looks only 2% higher, for a task such as legal entailment, it was a good boost. These results are higher than the best results on COLIEE leaderboard which was 28% F- measure. One thing I could infer using Graph CNN is, even if this has improved 2% results, there are very less false positive in the results as compared to previous methods. This way I can observe that less number of paragraphs detected as entailed where they were actually not. Graph CNN did show qualitative improvement in prediction of data.

After implementing Graph CNN on COLIEE, I followed the same implementation on SNLI and MNLI dataset to understand the potential of the model. In NLI challenge, I assigned unique IDs starting from 1 for each sentence pair. Using these IDs, I mapped premise and hypothesis together followed by extracting the vocabulary for entire dataset. The node size of MNLI is 754959 where as the node size of SNLI was 687994. Considering the training samples in both the datasets are large and so is the network, it took 4 days for the network to converge. In this experiment I created 4 hidden layers and input output layers architecture. The learning rate was set to 0.02, weight decay 0, 0.6 dropout. I trained the network for 600 epochs. The best results obtained on SNLI was 54% F-score. Noticeably, LSTM does give higher performance than this however, it would be interesting

to use word embeddings instead of one-hot encoded text in feature matrix to improve the performance of NLI task.

4.4 Evaluation Metrics

We use Precision, Recall and F-score [Goutte & Gaussier (2005)] to evaluate the proposed methods. Considering this is a classification task and one of the dataset is extremely skewed in distribution, accuracy would have been an unfair comparison. Think of an incidence where a classifier outputs all the classes as class "0" by simply discarding class "1" as noise, it will still show 97% accuracy as our data has 97% class 0. Due to this reason we selected precision recall and more importantly an f-score for performance evaluation. For COLIEE task we focused on class "1" results for all three measures considering the challenge was to find the decision of the case basis entailment. Here, having class 0 correctly classified was trivial. On the other hand, for SNLI and MNLI, I examined the performance of class 'entailed', 'contradiction' and 'neutral'. The motive behind not examining only entailment aspect of NLI challenge while we compare it with COLIEE is that, these datasets are used by researchers to establish contradiction as well. For instance, if we have a movie review which express half positive and half negative sentiment then detecting such contradiction is important to accurately understand what the user is saying. For such task, choice of training data has been MNLI. It is hence important to examine if these features and models can be used for extending the language tasks in other domain of Natural Language Processing.

4.4.1 Precision

In general precision is defined as "specificity". This is a true negative rate.

Mathematically it can be defined as

$$Specificity\ (TNF) = \frac{True\ Negative}{True\ Negative + False\ Positive}\ .$$

In the context of COLIEE, we will define precision as:

$$Precision = \frac{Number\ of\ correctly\ retrieved\ cases\ (paragraphs) for\ all\ queries}{Number\ of\ retrieved\ cases\ (paragraphs) for\ all\ queries}\ .$$

4.4.2 Recall

Recall is also known as "Sensitivity" measure. This specifies True Positive rate. It indicates how many cases are true positive and has been correctly detected as positive from entire sample size. Mathematically it can be defined as

$$Sensitivity\ (TPF) = \frac{True\ Positive}{True\ Positive + False\ Negative}\ .$$

In the context of COLIEE where we would examine how many cases were correctly obtained as entailed or not entailed and which were truly entailed and not entailed, we define recall as

$$Recall = \frac{Number\ of\ correctly\ retrieved\ cases (paragraphs) for\ all\ queries}{Number\ of\ relevant\ cases\ (paragraphs) for\ all\ queries}.$$

4.4.3 F-score

This is a weighted average of precision and recall and can be computed as:

$$F - score = \frac{2 * (Precision * Recall)}{Precision + Recall}\ .$$

Using F-score, we can truly measure a balanced performance of both the classes and ensure that we have realistic idea about how the model performs.

CHAPTER 5: RESULTS & DISCUSSION

I accomplished our entailment task on COLIEE, Canada Case Law dataset where designed models could deliver a decision for a given case with best performance using Topological data as well as graph CNN. Each method was then validated on SNLI and MNLI dataset. We will discuss our results for each chapter and on all three datasets along with preliminary observations here. COLIEE performance is compared with [Robelo et. al (2018)] who produced the best results on leaderboard. The results of SNLI and MNLI are compared with [Chen et al. (2018)] who achieved it using Enhanced Sequential Inference Model (ESIM) neural network. Unfortunately, we do not have their precision recall and F-measure for all three classes. Here we considered accuracies.

Table 5-1: Baseline results for SNLI and MNLI for further comparison with our methods.

Dataset	Accuracy
SNLI	88.0%
MNLI	75.8%

5.1 Similarity Relevance and TFIDF scores

Table 5-2 Results of COLIEE, SNLI and MNLI for similarity, relevance and TFIDF vector model. The classification is performed using Random Forest classifier.

Class	Precision	Recall	F-Score
COLIEE			
Robelo et al.,(2018)	0.24	0.28	0.26
Entailment	28.2	58.3	37.6
SNLI			
Contradiction	0.58	0.60	0.59
Entailed	0.60	0.57	0.59
Neutral	0.58	0.59	0.59
Micro avg	0.59	0.59	0.59
MNLI			
Contradiction	0.56	0.49	0.52
Entailed	0.53	0.30	0.38
Neutral	0.40	0.65	0.50
Micro avg	0.47	0.47	0.47

Cosine similarity along with relevance explained an entailment better than top results on COLIEE leaderboard. In this method, calculating only similarity between two documents and using it as a part of feature vector was not entirely enough to establish inference with stronger results. Cosine similarity captured relative angle between two documents which is independent of their length. BM25 algorithm was useful to find relevant noticed cases from the pool of documents given base case as a query. Using this technique when chosen most similar and relevant documents for further classification task to establish entailment, our algorithm performed better and could predict a

greater number of test case as correctly entailed. Here the focus is on F-measure to ensure less false positive entailment predictions.

For SNLI and MNLI, considering many examples have completely different terms in premise and hypothesis, we do not find cosine similarity and relevance explaining more than 60% of relations. BM25 works efficiently when used for finding relevant documents from collection of multiple documents. In SNLI and MNLI when incorporated pairwise relevance score for premise and hypothesis, it could not add value in increasing the results. MNLI is much harder to solve as compared to SNLI due to the complexity and length of the sentences. We would be able to conclude that although similarity and relevance works the best for document level entailment, neural networks could be a better choice to solve NLI task.

5.2 TFIDF vectors with topological features

Table 5-3 Results obtained on COLIEE, SNLI and MNLI using Topological features along with TFIDF vectors using Random Forest classification.

Class	Precision	Recall	F-Score
COLIEE			
Robelo et al.(2018)	0.24	0.28	0.26
Entailment	30.7	72.5	43.0
SNLI			
Contradiction	0.43	0.41	0.42
Entailed	0.31	0.62	0.41
Neutral	0.00	0.00	0.00
Micro avg	0.35	0.35	0.35
MNLI			
Contradiction	0.29	0.06	0.10
Entailed	0.40	0.94	0.56
Neutral	0.00	0.00	0.00
Micro avg	0.23	0.33	0.22

I achieved my first breakthrough of 43% F-score using topological features with TFIDF vector for classification purpose. Overall, topological signals are stronger when corresponding text pair is similar. Performance of TDA on SNLI and MNLI was inconsistent to my surprise where it cannot explain any neutral relationship at all. These features are very helpful in capturing entailment. It would be unfair to evaluate the performance of TDA on NLI task as training on the entire data was computationally expensive. Computation of TDA on the entire dataset remains my future work. Detail analysis of topological results on COLIEE is discussed in section 5.4.

5.3 Graph CNN for Entailment

Table 5-4 Results obtained on COLIEE, SNLI and MNLI using Graph CNN deep learning algorithm

Class	Precision	Recall	F-Score
COLIEE			
Robelo et al.(2018)	0.24	0.28	0.26
Not-entailed	0.50	0.56	0.53
Entailed	0.36	0.30	0.33
Micro avg	0.45	0.45	**0.45**
SNLI			
Contradiction	0.58	0.45	0.51
Entailed	0.53	0.57	0.55
Neutral	0.51	0.59	0.55
Micro avg	0.54	0.54	0.53
MNLI			
Contradiction	0.38	0.29	0.33
Entailed	0.26	0.27	0.27
Neutral	0.20	0.25	0.23
Micro avg	0.27	0.27	0.27

Observing my first breakthrough using TDA and examining the significant details topology has captured, I experimented with Graph CNN for establishing entailment. This method achieved our highest results of 45% F-measure on COLIEE data. Considering COLIEE evaluates on Micro average performance, I considered that number to evaluate the performance of this model. As compared to topology, although we see only 2% increment in Graph CNN, I would like to address

that this algorithm achieved balanced precision recall and f-measure. When dealing with class imbalance, one can achieve 100% recall by naively predicting all samples as major class, but this compromises the overall quality of prediction. Using Graph CNN, I achieved 45% balanced F-measure and my best results for COLIEE.

On SNLI and MNLI Graph CNN achieved 53% and 27% F-measure. We discuss the limitations and behavior behind this in the next section. One of the most powerful ability of this approach is to be able to generalize the data well. On observing SNLI training accuracy of 62%, one can conclude that graph can generalize well. Also, for fair comparison we trained the graph on full dataset however, graph can learn patterns on 1% data and perform equally well on the test dataset.

5.4 Discussions

Results mentioned in the previous section show that topological signals are stronger when given longer documents as an input. Also, for Graph CNN, the network performs better when computed on longer and more complex documents such as COLIEE data and previously on 20 news group dataset by Yao et al., (2018) as compared to SNLI and MNLI. While constructing a graph for shorter documents, there are less chances of capturing both the words $i \, and \, j$ together in convolving window. This results in negative pointwise mutual information between words of the shorter document. As discussed in theory, negative PMI result in poor edge structure. As the graph propagates label information through nodes via edges, not many vocabulary nodes receive document's label details and such architecture suffers from poor results in classification task. Intuitively one can say that for shorter and limited text, we can use pre-trained word embeddings such as word2vec or GloVe and try to make the text representation better. The graph learns predictive embeddings and does not learn from pre-trained embeddings. Taking this learning mechanism into consideration, one could stick to simple one-hot representation of the data. Another reason for graph to underperform on NLI task is, CNN does not consider the sequence in text. Due to the change in word order, the meaning of the entire sentence can change. Thus, this architecture suffers from learning entailment on NLI dataset.

Considering TDA gave a breakthrough in result by establishing structure between entailed document, we discuss an example case to demonstrate topological relationship in text. In this research, one of the main challenges is to find text behind topology to be able to infer what exactly topology has learnt while establishing entailment. To identify the relationship between entailed text document, I experimented with another TDA library GUDHI [. This allowed discovering persistent pairs of text. This experiment explains the topological structure formed between the documents

with exact values. After applying on one of the COLIEE case, it helped identifying the role of topology in inference.

Topological structure analysis for COLIEE test case:

Example case taken from COLIEE test data:

In this case an immigrant who has 2 children born in USA and 2 in Canada, decides to file for PR and get a citizenship in Canada to maintain lawful status of his kids and for his family. He is a refugee who came from another country to settle in Canada and has lack of documents due to circumstances. Due to these issues, the immigration officer is denying his application of PR and thus he went to the court.

Decision: In my view, it is important not to divorce the parties' submissions from the content of the application that was before the officer. Whatever the basis of an application under subsection 25(1) of the Act , public policy or otherwise, I am satisfied that the burden of producing proof of the claim is at all times upon the applicant.

Summary: Aguilar Espino and his family (the applicants) applied for permanent residence from within Canada on humanitarian and compassionate grounds (Immigration and Refugee Protection Act, s. 25(1)). An immigration officer dismissed their application. They applied for judicial review.

Entailed paragraph 1: An immigration officer considering an H & C application must be "alert, alive and sensitive" to, and must not "minimize", the best interests of children who may be adversely affected by a parent's deportation: Baker v. Canada (Minister of Citizenship and Immigration), [1999] 2 S.C.R. 817; 243 N.R. 22, at para. 75. However, this duty only arises when it is sufficiently clear from the material submitted to the decision-maker that an application relies on this factor, at least in part. Moreover, an applicant has the burden of adducing proof of any claim

on which the H & C application relies. Hence, if an applicant provides no evidence to support the claim, the officer may conclude that it is baseless.

Entailed paragraph 2: Mr. Owusu now says that while he has been in Canada he has supported his children, who are financially dependent on him, and that he has evidence to show that he has remitted money to them on a regular basis. Unfortunately, none of this was before the immigration officer when she made her decision. Apparently, Mr. Owusu's lawyer thought the grounds on which the H & C application was primarily based would be sufficient to obtain a favorable decision and that, in any event, Mr Owusu would be called for an interview at which he could present material showing that he had been supporting his children.

Analysis: Here we see that the case is about a person who applied for PR but his case has been refused by an officer. The first entailed paragraph derives the judgement support for why officer must have behaved like that. Second paragraph gives some more context as to why the applicant cannot provide paperwork, etc. These are the two entailed paragraphs using which, a decision can be made. Considering they mention *office should be "alert, alive and sensitive"* it is a big signal. Also: *Hence, if an applicant provides no evidence to support the claim, the officer may conclude that it is baseless.*

Predictions: Here the similarity alone cannot establish entailment as this is a deep semantic context. Below is the cosine similarity matrix where row 0 is decision, 1 is summary, 2 is facts, 3 is entailed paragraph 1 and 4 is entailed paragraph 2 data.

Cosine similarity matrix:

Table 5-5 The above table shows cosine similarity matrix calculated on one case. The example is taken from COLIEE 2018 dataset. This is an N by N matrix which shows cosine angle between decision, summary, fact and two of the entailed paragraphs for a case. The bold values show the topological ties where cosine similarity is not the highest.

	Decision	Summary	Fact	Entailed Para 1	Entailed Para 2
Decision	1	0.1854	0.7218	0.6369	0.3081
Summary	0.1854	1	0.4146	0.3242	0.2274
Fact	0.7218	0.4146	1	0.6585	0.5023
Entailed Para 1	**0.6369**	0.3242	0.6585	1	0.4256
Entailed Para 2	0.3081	**0.2274**	0.5023	0.4256	1

For bold values in similarity table, although they are not highest cosine similarity scores, they are the ones which contribute towards establishing entailment through topology. When we compute topological features on cosine distance matrix of these data points, we obtain persistence pairs in 5.1 [([3], [3, 0]), ([1], [1, 0]), ([4], [4, 1]), ([0], [3, 2]), ([2], [])] . Here 0 represents decision of a case, 1 represents summary, 2 shows fact and 3rd and 4th entry is for entailed paragraph 1 and 2.

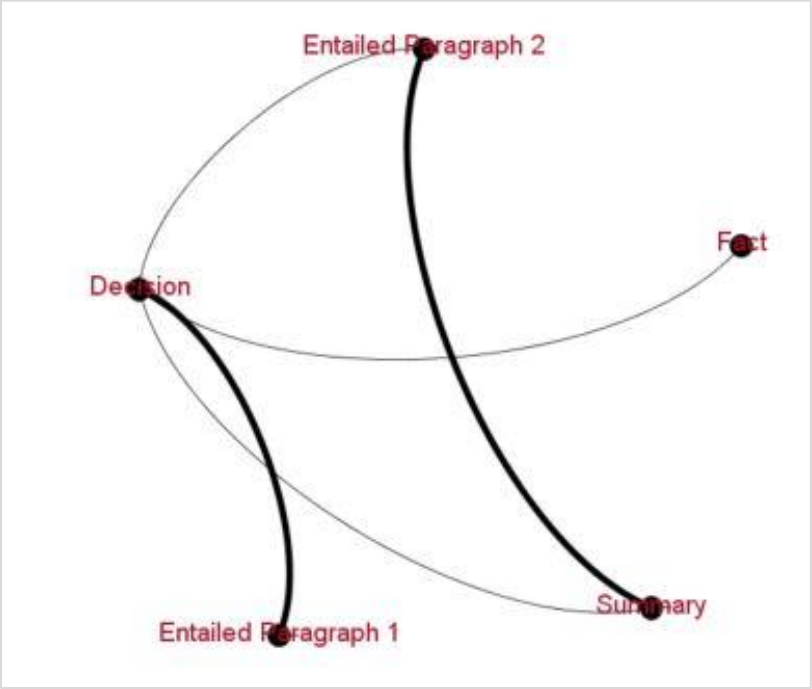

Figure 5-1 The above figure explains the connectivity of graph using topological features. The blue arrows show the ties from decision to other files and the orange arrows show how entailed paragraphs are linked with summary and decision. The ties back explain entailment better than similarity only.

The very first intuitive observation is, since the length of a text available in decision, summary, fact and paragraph files is relatively small, there are no loops available here. Although we do not have 1-D connected components, only dimension 0 can explain the topological relationship. As stated in figure 5-1, there is a connectivity of decision with 2 entailed paragraphs explaining why officer was correct in his way of behavior and why the applicant should get a benefit of his circumstances. Examining the values of cosine similarity in table 5-5, one can infer that cosine similarity alone cannot explain such relationship in given text. In the similarity table 5-5, we can observe 4th data point which is an 'entailed paragraph 2', is highly similar with FACT. However, topology shows ties with summary (index 1) file. This connectivity is created by persistence homology since topology is able to pick up the relationship between the two. The edges between the connected nodes are indicated with darker color and appear thicker. This shows that entailment is beyond

similarity *only* and requires deeper understanding to infer meaning of the text. The point of conclusion here is, topological features can capture semantic similarity in-between the documents beyond distance measure. Such information is valuable for machine learning classifiers to learn from.

CHAPTER 6: CONCLUSION & FUTURE WORK

In this research I have mainly focused on exploring unconventional feature representation for establishing entailment. The main purpose was developing an NLI model for legal documents (COLIEE 2018). Our model could provide a decision for a case from set of noticed cases. The primary features were similarity and relevance which explains how similar and relevant the set of references are from which we can further deliver the actual decision of a case based on entailment. We also explored geometry of text using topological data analysis and persistent homology. capturing the connectivity between documents through topology is a challenging area. It would be interesting to work on finding the exact text segment behind topological barcodes to leverage this technique with more reasoning. Although our breakthrough results were obtained using topology, we could improve them further with graph CNN, deep learning architecture. Graph CNN proves superior semantic understanding of language to capture entailment, especially for longer documents.

6.1 Future work

Computing homology on entire SNLI and MNLI dataset to get the better understanding and potential of topology for text requires enormous computing and memory resources. Developing computationally efficient and memory-safe algorithms for homology computation will be our natural next step. Being able to detect the exact segments behind topological barcodes would be one of the most exciting challenge and research area I have discovered through this thesis project. To test and verify the contributions of natural language inference on diverse NLP tasks, we will place the proposed features and model weights in the pipeline of Information Retrieval, Question Answer task in NLP. It would be interesting to observe the impact of entailment while improving language understanding of machines.

REFERENCES

[Caro et al., (2016)] Adebayo, K. J., L. Di Caro, G. Boella, and C. Bartolini. TEAMNORMAS's participation at the coliee 2016 bar legal exam competition, (submission id: N01). In Tenth International Workshop on Juris-informatics (JURISIN). 2016

[Baez, J. and M. Stay (2010)] Physics, topology, logic and computation: a rosetta stone. In New structures for physics, pp. 95–172. Springer. 2010

[Bankova et al., (2016)] Bankova, D., B. Coecke, M. Lewis, and D. Marsden. Graded entailment for compositional distributional semantics. arXiv preprint arXiv:1601.04908. 2016

[Baroni et al., (2012)] Baroni, M., R. Bernardi, N.-Q. Do, and C.-c. Shan Entailment above the word level in distributional semantics. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pp. 23–32. Association for Computational Linguistics. 2012

[Bauer, U. (2016)]. Ripser. https://github.com/Ripser/ripser. 2016

[Buitinck et al., 2013] API design for machine learning software: experiences from the scikit-learn project. 2013

[Bird et al., (2009)] Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc. 2009

[Bubenik, P. and T. Vergili (2018)] Topological spaces of persistence modules and their properties. Journal of Applied and Computational Topology, 1–37. 2018

[Gautte & Gaussier (2005)]]Cyril Goutte , Eric Gaussier, A probabilistic interpretation of precision, recall and F-score, with implication for evaluation, Proceedings of the 27th European conference on Advances in Information Retrieval Research, March 21-23, 2005, Santiago de Compostela, Spain [doi>10.1007/978-3-540-31865-1_25].2005

[Degan et al. (2010)] Dagan, I., B. Dolan, B. Magnini, and D. Roth (2010). Recognizing textual entailment: Rational, evaluation and approaches–erratum. Natural Language Engineering 16(1), 105–105. 2010

[Doshi, P. and W. Zadrozny (2018)] Movie genre detection using topological data analysis and simple discourse features. In Proc. 6th International Conference on Statistical Language and Speech Processing, SLSP 2018, Vol. 11171 of Lecture Notes in Computer Science, Springer.2018

[Edelsbrunner, H. and J. Harer (2010)] Computational topology: an introduction. American Mathematical Soc. Edelsbrunner, H., D. Letscher, and A. Zomorodian (2000). Topological persistence and simplification. In Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on, pp. 454–463. IEEE. 2010

[Gholizadeh, S., A. Seyeditabari, and W. Zadrozny (2018)] Topological signature of 19th century novelists: Persistent homology in text mining. Big Data and Cognitive Computing 2(4), 33.2018

[Guan et al., (2016)] Guan, H., W. Tang, H. Krim, J. Keiser, A. Rindos, and R. Sazdanovic. A topological collapse for document summarization. In Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on, pp. 1–5. IEEE. 2016

[Hennig, C. and T. F. Liao (2013)] How to find an appropriate clustering for mixed-type variables with application to socio-economic stratification. Journal of the Royal Statistical Society: Series C (Applied Statistics) 62(3), 309–369. 2013

[Hersh et al., (1994)] Hersh, W., Buckley, C., Leone, T., Hickman, D.: Ohsumed: an interactive retrieval evaluation and new large text collection for research. In: Croft, W.B., van Rijsbergen, C.J. (eds.) Proceedings of SIGIR-1994, 17th ACM International Conference on Research and Development in Information Retrieval, Dublin, IE, pp. 192–201. Springer, Heidelberg (1994)

[Hoang, Q. (2018)] Predicting movie genres based on plot summaries. arXiv preprint arXiv:1801.04813. 2018

[Horak et al., (2009)] Horak, D., S. Maleti´c, and M. Rajkovi´c Persistent homology of complex networks. Journal of Statistical Mechanics: Theory and Experiment 2009(03), P03034. 2009

[Huang at al., (2018)] Huang, H.-L., X.-L. Wang, P. P. Rohde, Y.-H. Luo, Y.-W. Zhao, C. Liu, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan (2018). Demonstration of topological data analysis on a quantum processor. Optica 5(2), 193–198. 2018

[Degan et al., (2005)] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognizing textual entailment challenge. In Proceedings of the PASCAL Challenges Workshop on Recognizing Textual Entailment, Southampton, UK. 2005

[Kipf & Wellings (2017)] Kipf, T. N., and Welling, M. 2017.Semi-supervised classification with graph convolutional networks. In ICLR. 2017

[Kim et al., (2016)] Kim, M.-Y., R. Goebel, Y. Kano, and K. Satoh (2016). Coliee-2016: evaluation of the competition on legal information extraction and entailment. In International Workshop on Juris-informatics (JURISIN 2016).

[Kerry Raymond (1989)] Kerry Raymond, A tree-based algorithm for distributed mutual exclusion, ACM Transactions on Computer Systems (TOCS), v.7 n.1, p.61-77, Feb. 1989 [doi>10.1145/58564.59295]. 1989

[Yao et al., (2018)] Liang Yao, Chensheng Mao, Yuan Luo 2018 : Graph Convolution Networks for text classification. arXiv : 1809.05679v3 [cs.CL] 13 Nov 2018

[Li B., Han L. (2013)] Li B., Han L Distance Weighted Cosine Similarity Measure for Text Classification. In: Yin H. et al. (eds) Intelligent Data Engineering and Automated Learning – IDEAL 2013. IDEAL 2013. Lecture Notes in Computer Science, vol 8206. Springer, Berlin, Heidelberg. 2013

[Niraj (2017)] Niraj Kumar https://www.linkedin.com/pulse/random-forest-algorithm-interactive-discussion-niraj-kumar/ . 2017

[Chen et al., (2018)] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In The 56th Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia. 2018

[Saikh et al., (2018)] Saikh T., Naskar S.K., Ekbal A., Bandyopadhyay S. (2018) Textual Entailment Using Machine Translation Evaluation Metrics. In: Gelbukh A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2017. Lecture Notes in Computer Science, vol 10761. Springer, Cham 2018

[Bowman et al., (2015)] Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. 2015. Recursive neural networks can learn logical semantics. In Proc. of the 3rd Workshop on Continuous Vector Space Models and their Compositionality. 2015

[Bowman et al., (2015)] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2015

[Harabagiu and Hickl. (2006)]. Sanda Harabagiu and Andrew Hickl. Methods for using textual entailment in open-domain question answering. In Proceedings of ACL, pages 905–912, Sydney, Australia. 2006

[Jane & Lee (2006)] Show-Jane and Yue-Shi Lee (2006) “Under-Sampling Approaches for Improving Prediction of the Minority Class in an Imbalanced Dataset” : SCI2S research group. 2006

[Robertson & Zaragoza(2006)] Stephen Robertson and Hugo Zaragoza (2009), "The Probabilistic Relevance Framework: BM25 and Beyond", Foundations and Trends® in Information Retrieval: Vol. 3: No. 4, pp 333-389. http://dx.doi.org/10.1561/1500000019 . 2006

[GUDHI] Research supported by Advanced Grant of the European Research Council GUDHI

[Rehurek & Sojka (2010)] Radim Rehurek, Petr Sojka (2010) Software framework for topic modelling with large corpora . THE LREC 2010 WORKSHOP ON NEW CHALLENGES FOR NLP FRAMEWORKS. Pages 45—50 , University of Malta. 2010

[Lior & O (2008)] Rokach, Lior; Maimon, O. (2008). Data mining with decision trees: theory and applications. World Scientific Pub Co Inc. ISBN 978-9812771711. 2008

[Khot et al., (2018)] T. Khot, A. Sabharwal, and P. Clark. Scitail: A textual entailment dataset from science question answering. In Proceedings of AAAI, 2018. 2018

[Sammut & Webb (2011)] (2011) TF–IDF. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. 2011