Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Computer Science for Digital Media

Learning to Paraphrase from Multi-Document Summarization Data

Master's Thesis

Thang Nguyen
Born Sept 10, 1991 in VietNam

Matriculation Number 119533

1. Referee: Prof. Dr. Benno Stein
2. Referee: PD Dr. Andreas Jakoby

Submission date: January 24, 2022

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, January 24, 2022

.....
Thang Nguyen

Abstract

A paraphrasing process is a set of algorithms for not only detecting, identifying, extracting paraphrases from a large dataset of texts, but also generating paraphrases for arbitrary input texts. Generally, in the former type of tasks, given an arbitrary text which can include sentences, paragraphs or even an entire document from a user, a paraphrasing process requires retrieving crucial information, then having it compared with the source text on the basis of defined metrics to decide whether they are paraphrased or not. For a text generation, a model (usually a machine learning or a deep learning model) learns to paraphrase arbitrary input texts from a huge dataset of paraphrases.

Tasks related to paraphrasing usually face two main challenges, namely the lack of labelled data and their complexity. Indeed, the growth of digital contents on platforms like social media and online forums results in a huge volume of documents, significant diversity of topics as well as contents. Notably, in order to use such data, each sample needs to be labelled manually. Additionally, at the moment, almost all paraphrasing-related datasets comprise short texts, specifically sentences, with simple structures. When models are trained with these datasets, the performance on longer texts such as complete paragraphs or documents is often low.

This master thesis combines several semantic similarity metrics and machine learning algorithms to create a dataset for paraphrasing longer texts, specifically paragraphs. This dataset is then used for training and fine-tuning a deep learning model of text generation, in which generated text and source text are paraphrases of each other. Then the performance of the model is evaluated by using both automatic evaluation (text similarity metrics) as well as crowd-based evaluation (annotators are asked to answer lists of questions).

Contents

Chapter 1

Introduction

Paraphrases are texts conveying the same meaning while using different words (Bhagat and Hovy [2013]). As discussed by Cao et al. [2017], Fader et al. [2014], Berant and Liang [2014], there are various applications of paraphrasing, for example text summarization, plagiarism detection, information retrieval and question-answering systems.

Paraphrasing is a very active and well-researched area of Natural Language Processing. For the last 2 years alone, based on *www.dblp.org*, there have been more than 200 publications on automatic paraphrasing, paraphrase recognition, paraphrase identification, and related topics. For example, Thompson and Post [2020] propose a new approach for a paraphrase generation method, based on penalizing overlapping n-grams between generated and source sentences. Sokolov and Filimonov [2020] utilize a translation model to develop a generator which is used to produce translated-and-paraphrased texts in many languages.

Previously, some research papers dealt with tasks of paraphrase generation, such as Cao et al. [2017], Prakash et al. [2016], Gupta et al. [2018b], Su and Yan [2017], while others investigated topics of paraphrase identification (Blacoe and Lapata [2012], Socher et al. [2011]). However, most of them are focused on short text and / or sentence level. Additionally, the level of paraphrasing is still basic, such as changing a random word in a sentence by its synonyms, or rearranging word orders, or switching between passive and active modes.

This thesis is aimed at going more deeply into paraphrasing with more focus on long texts, particularly paragraphs. To make it more specific, efforts have

been made to extract paragraphs which are paraphrases of one another, then use them as inputs for deep learning on the basis of paraphrase generation models.

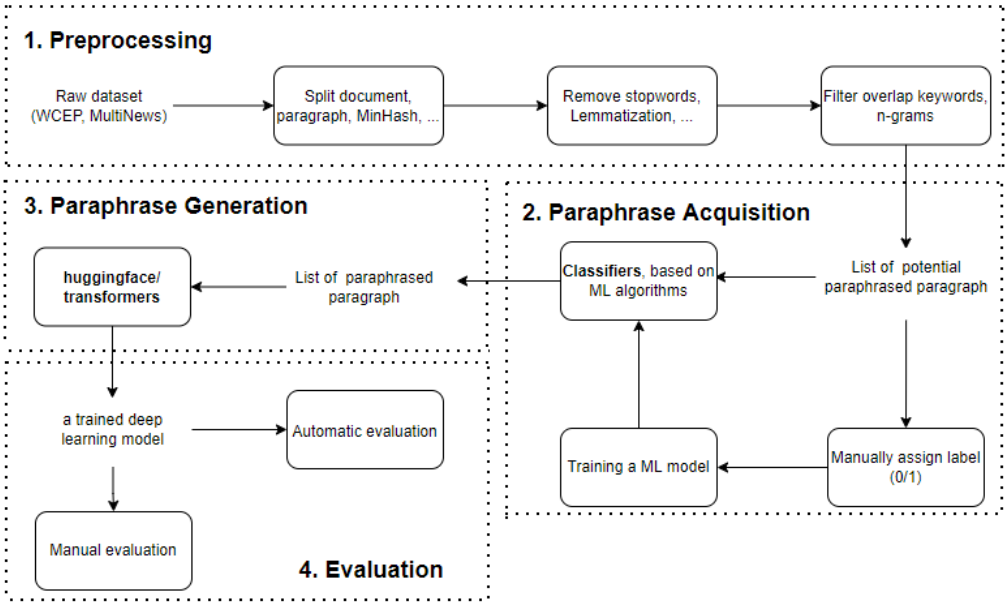

Figure 1.1: The general process of this thesis.

The overview of the whole process is visualized step by step as shown in Figure 1.1. The proposed approach is as follows:

1. We clean and preprocess data (**Multi-News**, created by Yale University, which consists of news articles & human-written summaries of these articles from the site *newser.com*, and **Wikipedia Current Events Portal**, collected by Ghalandari et al. [2020], which contains short, human-written summaries about news events) in multiple steps with redundant components reduced and crucial information extracted to optimize the subsequent actions.

2. Based on data from the previous cleaning and preprocessing step, we use various machine learning algorithms in parallel as classifiers to split paragraphs in collected data into 2 groups: paraphrased and not-paraphrased texts.

3. With data of paraphrased paragraphs, we train a deep learning model of paraphrase generation to automatically create a paraphrased text, given an arbitrary input text.

4. We use a number of manual as well as automatic methods to evaluate the

quality of output texts of the above deep learning model. They are assessed in many aspects, including the extent of keeping all main information in source text and the difference in grammatical structure between generated and source text.

The key contributions of this thesis include:

A new paragraph-level paraphrasing dataset, which contains approximately 6000 pairs of paragraphs. Accordingly, a paraphrase generator is trained and used to create paraphrased texts, given an arbitrary paragraph. The performance of this model is evaluated by a list of opinion questions answered by annotators.

The following chapters of this thesis are organized as follows: **Chapter 2** discusses related work with regard to approaches used to identify or create paraphrase corpus. It can be seen that research related to this topic can be split into 2 main branches: paraphrase identification and paraphrase generation. Notably, almost all prior works focus on short-level text, such as single sentences. The next chapter, **Chapter 3**, which can be deemed to be one of the most important parts of this thesis, describes and analyzes the multi-document summarization corpora that form the basis for this thesis. It also clarifies the whole process of extracting a list of potential paragraph-level paraphrases from several multi-document summarization datasets by using various data cleaning and data processing techniques. Then, **Chapter 4** describes how to train a deep learning model to create paraphrased paragraphs, given an input paragraph. There are 2 main phases: Training Phase and Testing Phase. Then a simple tool is introduced to use this model more easily to generate texts. To evaluate the performance of this model, this chapter describes the progress of evaluating the quality of the entire process. Comparisons are made between the source paragraph and generated paragraph (output of the deep learning model) among three variants of paraphrase-generation models in two different ways. The first is manual evaluation, where annotators are asked to judge the quality of paraphrases. The second is automatic evaluation, where semantic similarity metrics are used to compute precision, Recall, and F1-score for each pair of paragraphs. The findings reveal that human annotators tend to prefer texts generated by fine-tuning models. Meanwhile, automatic metrics are not really useful when discriminating them, when the difference in their mean value of Precision, Recall and F1-score is not significantly high. Finally, **Chapter 5** compares differences in structure and characteristics of paraphrasing at the sentence level which are often observed in most previous research papers, and that in paragraph level applied in this thesis, as well as limitations

on available datasets. Then, we discuss and explain difficulties and limitations of the results completed in this thesis. This is followed by recommendations for future improvements.

Chapter 2

Related Work

Paraphrase-related topics have attracted a lot of different NLP approaches. There are 2 main branches of Paraphrase topic. Firstly, Paraphrase Identification is applied when we want to recognize whether two texts share the same/similar meaning. Secondly, Paraphrase Generation can be used with a given arbitrary text, which could be a sentence or a long document. Thus, the model will generate a text which is the paraphrase of the input text.

For Paraphrase Detection/Identification, according to Mohamed and Oussalah [2020], research related to this topic is usually categorized into three high levels: corpus-based, knowledge-based, and hybrid methods. Firstly, corpus-based approaches can take advantage of corpus statistics to solve paraphrase identification/detection problems. In 2013, Ji and Eisenstein [2013] proposed a new discriminative term weighting metric called TF-KLD, since it derives from both words: the term frequency and the KL-divergence. Also, they argued that this metric could outperform TF-IDF which has been widely used. A year before that, Blacoe and Lapata [2012] combined three types of representations of texts: simple semantic space, syntax-aware space, and word embeddings. Another perspective is proposed by Wan et al. [2006], using lexical and syntactic dependency-based features to develop a machine learning model for paraphrases detection. In 2005, Finch et al. [2005] applied WordNet to develop a translator used for identifying paraphrases.

Secondly, in terms of knowledge-based methods, Fernando and Stevenson [2008] introduced a method by deriving WordNet to evaluate similarity level among texts. Also, WordNet, Das and Smith [2009] used a probabilistic model based on quasi-synchronous dependency grammars. A few years after that, by us-

ing information from Wikipedia, Hassan and Mihalcea [2011] proposed a new method called Salient Semantic Analysis (SSA). Furthermore, Kozareva and Montoyo [2006] introduced a new approach based on matched content among texts (e.g: e-grams and private name), as well as semantic features from WordNet.

According to Mohamed and Oussalah [2020], hybrid methods used at least two information sources, from simple things such as distributional statistics, path lengths between concepts in graphical knowledge representations, to more complex things like machine learning algorithms. Mihalcea et al. [2006] used the combination between corpus-based and knowledge-base using TF-IDF, based on WordNet and the British National Corpus. On the other hand, Qiu et al. [2006] and Wang et al. [2016] proposed a model of paraphrase identification, using not only the similarity but also dissimilarity between sentences. In another research, Islam and Inkpen [2008] introduced a sentence similarity model derived from the semantic and syntactic information.

The second branch, Paraphrase Generation, has also received significant attention and research. For instance, methods based on Retrieval-based text generation have been the targets for a lot of research for recent years. Song et al. [2016] and Wu et al. [2019] introduced new Seq2Seq generation-based models to improve the dialogue response quality. Meanwhile, Gu et al. [2017] trained a model of search engine to translate an input query. A year after that, Guu et al. [2018] proposed a neural editor model for text generation.

Thanks to the development of research into deep learning over the last decade, approaches based on neural networks have also been applied widely. Prakash et al. [2016] were one of the first researchers who used a residual stacked LSTM network to develop a paraphrase generator. Similarly, Gupta et al. [2018b] utilized the combination of a variational auto-encoder and a Seq2Seq LSTM model to generate paraphrases for a given input text.The year of 2019 saw many new paraphrased researches published. Kajiwara [2019] introduced a 2-steps model that first extracts a list of words needing paraphrasing, then generated the output by using a pre-trained paraphrase generation model. In another research, Wang et al. [2019] suggested improving the quality of paraphrases by using a Transformer-based model.

Finally, several datasets related to paraphrasing were widely used in previous research as illustrated in Table 2.1.

Table 2.1: Paraphrasing datasets in previous researches. Their crucial information is provided such as their names, how big they are.

Dataset name	Text type	Number of pairs	Citation
Paraphrase Adversaries from Word Scrambling	Sentences	725,450	Zhang et al. [2019b]
Paralex	Sentences	18,000,000	Fader et al. [2013]
Paraphrase and Semantic Similarity in Twitter	Sentences	18,762	Xu et al. [2015]
Quora Question Pairs	Sentences	404,290	Sharma et al. [2019]
Microsoft Research Paraphrase Corpus	Sentences	5801	Dolan and Brockett [2005]

Almost all datasets in Table 2.1 were used frequently in previous researches and they also were constructed in similar ways. For example, based on Dolan and Brockett [2005], Microsoft Research Paraphrase Corpus consists of thousands of pairs of sentences, each accompanied by a binary judgment indicating whether human raters considered the pair of sentences to be similar enough in meaning to be seen as close paraphrases. The general feature from all datasets in this table is that they are sentence-level paraphrasing datasets.

Chapter 3

Dataset Construction

Many datasets are to be analyzed throughout this thesis. However, we particularly deeply research and directly use only two of them. They are both unlabeled datasets, which are **MultiNews**, **Wikipedia Current Events Portal (WCEP)**.

3.1 Data Sources

We introduce general information related to MultiNews and WCEP such as their original published paper and their size.

3.1.1 MultiNews

This data set was first introduced in Fabbri et al. [2019]. MultiNews was originally designed for the task of multi-document summarization, when we want to produce a shorter version of one or several documents that preserve most of the input's meaning. It consists of news articles and human written summaries of these articles from the site of newser.com. Each summary is professionally written by editors. It is split into training, validation and test sets as shown in Table 3.1.

Table 3.1: Statistics related to **MultiNews** datasets. It shows the number of samples and proportions among 3 subsets.

MultiNews		
	Count	Proportion
Training set	44,972	80%
Validation set	5,622	10%
Test set	5,622	10%
Total	56,216	100%

Moreover, because the number of articles per sample is varied in this dataset, we have Table 3.2 for their distribution. It shows that over 80% of the summarised texts in MultiNews datasets are synthesized from less than 4 source texts. Furthermore, a lot of statistics related to the length of each article are also visualized in Table 3.3 (left). Finally, because the main research object in this thesis is paragraphs, a number of statistics related to their lengths are also demonstrated in Table 3.4 (right). Table 3.3 & Table 3.4 reveal that the number of paragraphs per article as well as the number of sentences per paragraph in this dataset is significantly diverse in general and that they do not focus on any specific value.

Table 3.2: The number of articles per sample in **MultiNews** datasets. It illustrates the number of input texts for each summarized text.

Num of Articles	**Frequency**	**Proportion**
2	29868	53.1%
3	15883	28.3%
4	6277	11.2%
5	2341	4.2%
6	954	1.7%
7	478	0.9%
8	261	0.5%
9	111	0.2%
10	43	0.1%

Table 3.3: The number of paragraphs per article in **MultiNews** datasets.

Par./Art.	**Count**	**Proportion**
1	9871	6.39%
2	6855	4.44%
3	4319	2.79%
4	4848	3.14%
5	4784	3.10%
6	4895	3.17%
7	5280	3.42%
8	6057	3.92%
9	5544	3.59%
10	5468	3.54%
>10	96623	62.52%
Total	154544	100.00%

Table 3.4: The number of sentences per paragraph in **MultiNews** datasets

Sent./Par.	**Count**	**Proportion**
1	30760	13.01%
2	18783	7.94%
3	16619	2.79%
4	9090	3.14%
5	6845	3.10%
6	5887	3.17%
7	5790	3.42%
8	5228	3.92%
9	5052	3.59%
10	5016	3.54%
>10	123875	52.38%
Total	236493	100.00%

3.1.2 Wikipedia Current Events Portal

Ghalandari et al. [2020] were the ones who first introduced the dataset of Wikipedia Current Events Portal. It consists of 10,200 clusters with one human-written summary and 235 articles per cluster on average. It is split as shown in Table 3.5. As shown in Table 3.5, this dataset is split in the same way as in MultiNews dataset.

Table 3.5: Statistics related to WCEP datasets. It shows the number of samples and proportions among 3 subsets.

WCEP		
	Num of Samples	Proportion
Training sets	8,158	80%
Validation sets	1,020	10%
Test sets	1,022	10%
Total	10,200	100%

Furthermore, Table 3.6 visualizes statistics related to the length of each article. It demonstrates that approximately 50% of the articles in WCEP datasets have less or equal to 10 paragraphs.

Table 3.6: The number of paragraphs per article in **WCEP** datasets.

Num of Paragraphs	Count	Proportion
1	33910	5.22%
2	24888	3.83%
3	20222	3.11%
4	39143	6.03%
5	45463	7.00%
6	37135	5.72%
7	31651	4.87%
8	30474	4.69%
9	30797	4.74%
10	30988	4.77%
More than 10	324926	50.02%
Total	649597	100%

Finally, because the main research object in this thesis is paragraphs, statistics related to their lengths are also shown in Table 3.7. It can be seen that the majority of the paragraphs in WCEP dataset are short ones, often in the range of one to three sentences. More specifically, more than 64% of the paragraphs in this dataset have only one sentence, and over 90% of the paragraphs have less than four sentences.

Table 3.7: The number of sentences per paragraph in **WCEP** datasets.

Num of sentence	Count	Proportion
1	5333680	64.4166%
2	1974749	23.8497%
3	565446	6.8291%
4	171175	2.0673%
5	64584	0.7800%
6	28628	0.3457%
7	14051	0.1697%
8	7451	0.0900%
9	4562	0.0551%
10	3171	0.0383%
More than 10	112485	1.3585%
Total	8279982	100%

3.2 Paraphrase Identification

A dataset of paraphrasing at paragraph-level is collected from multiple datasets.

3.2.1 Introduction

There are so many ways to define **"What a paraphrase is ?"** in the NLP literature. Madnani and Dorr [2010] define a paraphrase as an alternative surface from the same language expressing the same semantic content as the original form. Based on Pang et al. [2003], paraphrasing is the way to express the same information in multiple ways. Meanwhile, Ganitkevitch et al. [2013], argues that "paraphrases are differing textual realizations of the same meaning".

There are also many datasets used in related paraphrase topics. Usually, in their datasets, 2 types of labels are used: 1/positive pair for paraphrased texts and 0/negative pair for not-paraphrased texts.

In this thesis, the distinction between positive pairs (paraphrased) & negative pairs (not paraphrased) is identified based on features such as overlapping content words, trigrams as well as metrics such as BLEU, SUMO, Jaccard similarity, etc..

However, since there are numerous methods, and their performance is also different, selecting suitable metrics/methods in this thesis is not straightforward. Moreover, while most of the above-mentioned metrics/methods are used with short texts like sentences, the main objects in this thesis are long and complex structured texts such as paragraphs. Therefore, the adopted approach is to apply different metrics/methods on a labeled dataset such as Microsoft Research Paraphrase datasets (MRPC), then evaluate their performance and find out a way to combine them together to achieve the best performance.

3.2.2 Preprocessing

In the datasets of **MultiNews**, articles of a sample as identified by ||||| characters and paragraphs are defined by NEWLINE_CHAR NEWLINE_CHAR characters.

In the datasets of **WCEP**, the structure of each sample is reflected in Figure

3.1. It shows that the "summary" element is summarized news, synthesized from two or more articles embedded in the "articles" element.

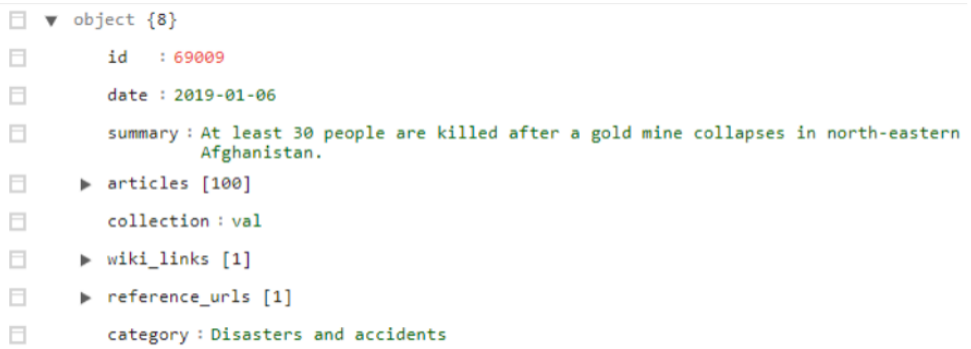

Figure 3.1: Structure of each sample in WCEP datasets. Contents of articles are in the "articles" element.

In both datasets, each article is split into paragraphs. Redundancy such as financial or advertisement paragraphs are removed, because they are not the targets for this project on paraphrasing research. Moreover, too short paragraphs, which contain only 1 sentence, and too long paragraphs, which contain more than 6 sentences are also skipped. Notably, this project is mainly targeted at paragraphs. Meanwhile, paragraphs which consist of only one sentence can be subject to sentence-level analysis. In contrast, the longer paragraphs are, the less likely they contain content equivalent to that in other articles. When manually testing several times in both MultiNews & WCEP, I see that usually, when the length of a paragraph is over 6, it is difficult to find its paraphrase in other articles. Therefore, to optimize searches and analysis in this project, it is arguably necessary to focus on paragraphs which are average in length, i.e. in the range of 2 to 6 sentences, etc..

In each paragraph, the following data preprocessing methods are applied:

Lemmatization: It helps us to achieve the root forms of each word. Languages we speak and write are made up of so many words, which are often derived from one another. Then it is called Inflected Language.

In grammar, inflection is the modification of a word to express different grammatical categories such as tense, case, voice, aspect, person, number, gender, and mood. An inflection expresses one or more grammatical categories with a prefix, suffix or infix, or another internal modification such as a vowel change. (Wikipedia [2022])

The rationale to apply Lemmatization lies in its usefulness in reducing the size of the dictionary. It can minimize the data space and the need to check every single form of a word.

For example, given a collection of documents, all documents in which anything related to eating is mentioned should be collected. Thus, words such as eat, ate, eaten must be searched. In another example, in developing a search engine, a lot of useless and ugly codes have to be written to handle a seemingly non-exhausting number of cases. By using Lemmatization, we can convert all words into their root words.

Stopwords: which are the most common words such as you, we, they, is, was, will, ... are dropped. This is because most search engines avoid them for the purpose of focusing on the important words.

3.2.3 Feature Engineering for Paraphrase Identification

There are 7 different features used in the process of distinguishing between paraphrase & not-paraphrase texts in our labeled dataset (MRPC). They are:

Content word: The number of overlapping content words between 2 texts (content words are those which contain useful information such as name, date, time, number).

Single word: The number of overlapping common words between 2 texts (stopwords are excluded).

Tri-grams: The number of overlapping trigrams between 2 texts.

Jaccard similarity: The measure of similarity for 2 texts. It is the ratio between the size of the intersection and the size of the union of two sets being compared. In this case, common words, which appear in two texts, are contained in two sets.

BLEU: An algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. It can also be used for evaluating paraphrasing texts. It compares the n-gram of the source text with that of the generated text to count the number of matches. (Papineni et al. [2002])

SUMO metric: a new metric for unsupervised detection of paraphrases. It is computed based on not only the number of overlapping words but also the

length of two texts. (Cordeiro et al. [2007])

Sentence embedding: Multilingual Sentence Embeddings using BERT & RoBERTa & XLM-RoBERTa & Co. with PyTorch. (Reimers and Gurevych [2019])

When applying all of the 7 metrics/methods into positive & negative pairs in the labeled dataset (MRPC), we have results visualized via Figure 3.2, while their mean & standard deviation are shown in Table 3.8.

	Negative pairs	Positive pairs
Content Words		
Mean	1.07207	1.27184
Std	1.02958	1.13800
Single Words		
Mean	6.68595	8.95555
Std	2.84082	3.42626
Tri-grams		
Mean	2.54182	4.01670
Std	2.52394	3.17743
Jaccard Similarity		
Mean	0.39644	0.55094
Std	0.14995	0.17716
BLEU		
Mean	0.50219	0.64802
Std	0.11855	0.14122
SUMO		
Mean	3.15389e-05	6.60510e-05
Std	3.97089e-05	6.45772e-05
Sentence Embedding		
Mean	0.63095	0.83047
Std	0.18326	0.12490

Table 3.8: Mean and standard deviation for positive & negative pairs of 7 metrics.

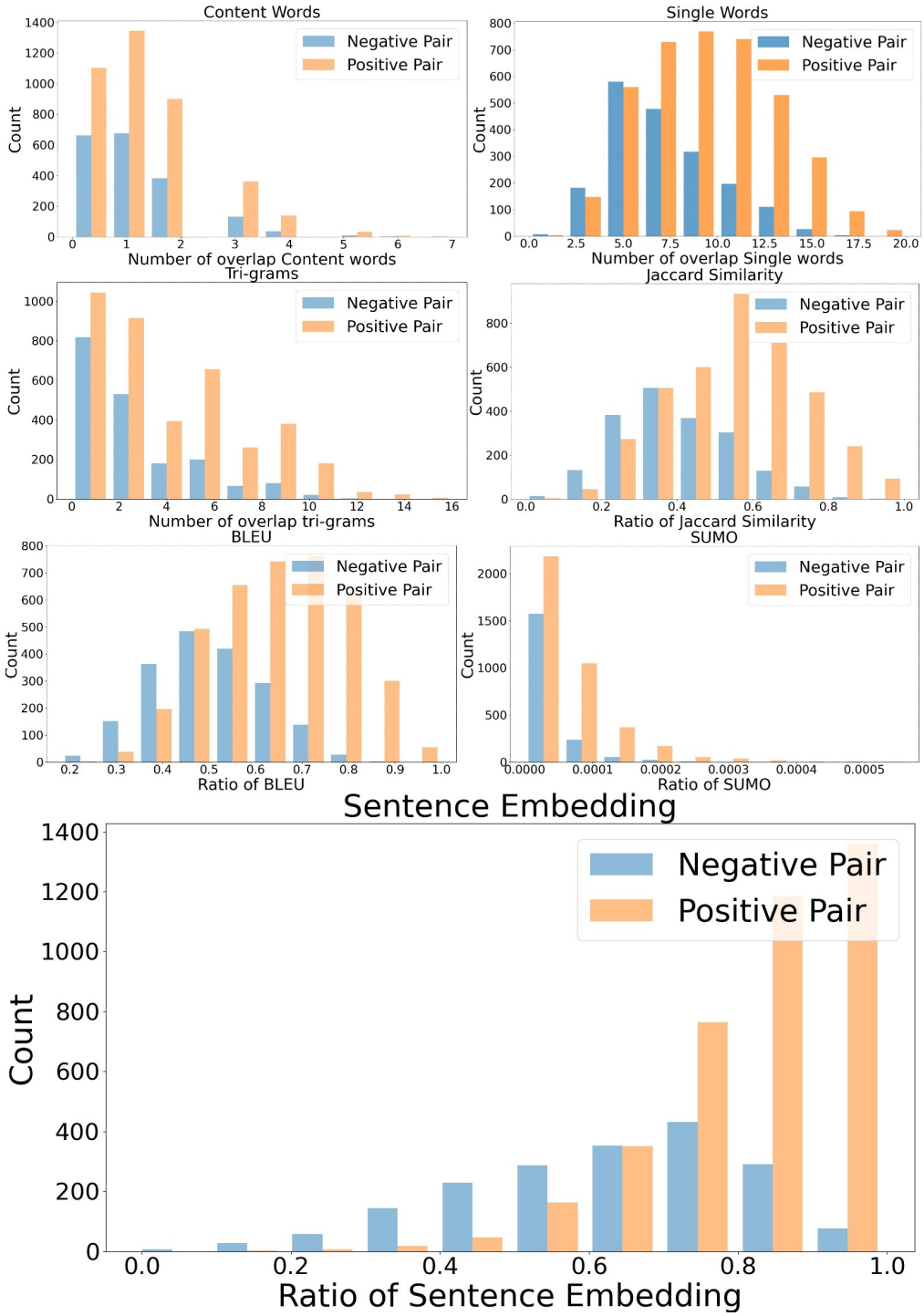

Figure 3.2: Histogram of 7 methods between positive & negative pairs.

Based on the histograms for the distribution of values as well as the table for mean & standard deviation values, it is evident that the average values of positive pairs are usually higher than negative ones among all 7 features.

In this thesis, these 7 methods are divided into two groups to be used:

Group 1 - Simple metrics/method: Content words, Single words, Trigrams, Jaccard Similarity. All metrics in this group are only based on simple formulations to evaluate similarity between pairs of texts. Hence, they are used in the first steps of our process, to remove obviously not-paraphrased text.

Group 2 - Complex metrics/method: BLEU, SUMO, Sentence Embedding. Their evaluation is based on more complex calculations to analyze more deeply the similarity between elements. So, they are used in later steps of our process, when we want to exactly filter paraphrased text.

3.2.4 Paragraph-level Paraphrase Identification

The process of paraphrase identification in this thesis is structured into 2 main steps as follows:

3.2.4.1 Step 1 - Filtering obviously not-paraphrased text

In this step, features in Group 1 are used to identify obviously not-paraphrased pairs of paragraphs in 2 unlabeled datasets: **MultiNews** and **WCEP**.

Firstly, all articles in each sample are grouped into a list of article pairs. In each pair of articles, their meaning similarity is calculated based on Jaccard Similarity. MinHash is a technique which is used to quickly estimate their similarity. The idea behind MinHash is representing each article as a signature. A signature preserves a permutation of a bit array representation of an article. By using hash functions that simulate a permutation, the probability of collisions against all permutations results to the Jaccard similarity. (Broder [2000])

The reason why MinHash is used is that, regarding combinations, when the number of elements increases linearly, the number of their combinations will increase exponentially.

For example: Given 5 articles, there are 10 different ways to combine 2 arbitrary articles into a pair;

$$C_2^5 = \frac{5!}{2!(5-2)!} = 10 \tag{3.1}$$

However, when the number of articles increases by ten times to 50 articles, there are 1225 different ways to combine them into pairs;

$$C_2^{50} = \frac{50!}{2!(50-2)!} = 1225 \tag{3.2}$$

In the dataset of WCEP, usually there are from 50 to 150 articles per sample. Actually, in the dataset of WCEP alone, there are over 110 millions possible article pairs. If we used normal ways to calculate their semantic similarity, it would be time and memory consuming. After the MinHash technique is applied, all pairs of articles with Jaccard Similarity of less than 0.3 or more than 0.7 will be removed.

If a pair of articles is of too low Jaccard similarity, its probability for some texts to be paraphrases of each other might be very low as well. Also, it should be noted that, though all articles are from the same sample of datasets (WCEP or MultiNews), i.e., all articles are about the same topic, and/or the same event, if the Jaccard similarity is too low, it is considerably likely that each article is about a different aspect of that topic and/or event. That results in the contents of articles to be different, and leads to the absence of texts which are paraphrases of each other.

On the other hand, if the Jaccard similarity value of a pair is too high, it is significantly likely that the contents of that pair are (nearly) identical. The reason is that most of these articles are from the same source of news, leading to the fact that the number of overlapping parts is also high. Therefore, if the similarity of a pair of articles is too high, they should be dropped.

After testing by many pairs of threshold values in both datasets (MultiNews & WCEP), the pair of (0.3 & 0.7) is a reasonable value for it (MinHash). For the higher threshold, from my observation in many articles of both WCEP & MultiNews datasets, it is usually that when the jaccard similarity between a article pair is over 0.7, whether they have many overlap paragraphs to each other, or their contents are nearly duplicated to each other (only difference in several minor information). On the other hand, for the lower threshold, when the jaccard similarity is lower than 0.3, the contents in these articles are usually very different to each other, leading to difficult to find paraphrased paragraphs in both articles.

After Step 1, approximately 85% of the article pairs with a Jaccard Similarity value that is too low or too high would be removed.

3.2.4.2 Step 2 - Finding potential paraphrasing paragraphs

This step is expected to search among all paragraphs in each article pair to find which paragraph pairs are potential paraphrasing paragraphs. When a pair of paragraphs is considered, all (nearly) duplicated sentences are skipped to guarantee the quality of the comparison. They are detected based on Levenshtein edit distance.

Then, all content words in each paragraph are extracted and compared to each other to find common/overlapping content words.

Content words can be defined as words which provide concise representations of the contents of the article being considered. Content words also help us locate the article from information retrieval systems. Accordingly, in this project, content words have an important role in deciding if a pair of paragraphs is considered a potential paraphrase or not.

Based on the overlap of content words between pairs of paragraphs, paragraphs which are clearly not paraphrases of each other are filtered out. In other words, pairs of paragraphs which do not contain any overlapping content words, and therefore are not about the same topic or event, are excluded.

Next, Jaccard Similarity is used to evaluate the similarity between the two paragraphs. Similarly to the approach at the article level, pairs of para-graphs with too high or too low Jaccard similarity are dropped. Based on my observation, when the jaccard similarity between a paragraph pair is lower than 0.3, their contents are rarely paraphrases of each other. On the contrary, when the contents of paragraph pairs with their jaccard similarity to be higher than 0.7, usually they have many identical sentences. Finally, overlapping ngrams (specifically here tri-grams) are also searched in order to filter out pairs of paragraphs with low probability of paraphrasing. N-grams such as bi-grams or tri-grams are also widely used as lexical features in previous research papers related to Paraphrase Identification such as Dey et al. [2016].

Notably that, based on the histogram of jaccard similarity between positive & negative pairs in Figure 3.2, it seems to be that our chosen thresholds (0.3 and 0.7) are unreasonable, when almost every pair with jaccard similarity higher than 0.7 is a positive pair. However, while this histogram is calculated on

single-sentence objects, our target in this thesis is on paragraph level, so these threshold values are not the same to each other.

All pairs of paragraphs which pass all above-mentioned steps will be the ones with high probability of being paraphrases of each other. The number of potential paraphrase pairs before and after removing duplicated pairs is 70,185 & 21,773, 35,455 & 13,410 in WCEP and MultiNews, respectively.

3.2.4.3 Step 3 - Improving results

There is a list of over 35 thousand potential paraphrase pairs in both datasets (MultiNews & WCEP). However, when several pairs are randomly picked from this list to review manually, they are usually not paraphrases of each other. It means that metrics such as content word, single word, tri-grams, jaccard similarity are not sufficient to identify paragraph-level paraphrases.

Hence, other metrics such as BLEU, SUMO should be used as additional features to improve the quality of this process. Notably, while Jaccard similarity is used in Step 1 as a filter to remove obvious not-paraphrase text, it is used in Step 3 as a feature, to train Machine Learning algorithms in Step 4:

The performance of these features is evaluated by applying them into the labeled datasets (MRPC) and they are illustrated in Figure 3.2 and Table 3.8. Moreover, in order to have better comparison of their performance, AUC-ROC curve can be applied into each feature independently.

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all possible classification thresholds. This curve plots two parameters: True Positive Rate & False Positive Rate.

An AUC (area under curve) is a single scalar value in the range of [0.5 1.0] that measures the overall performance of a binary classifier. While this value equals to 0.5, it presents the performance of a random classifier model. It should be mentioned that a perfect classifier model corresponds to a maximum value of 1.0. AUC is a robust overall measure to evaluate the performance of a model because its calculation is based on a complete receiver operating characteristic (ROC) curve and AUC considers all possible classification thresholds.

In this experiment, AUC is used to compare the performance among 4 metrics: Jaccard similarity, BLEU, SUMO metric, Sentence Embedding in MRPC datasets. All positive and negative pairs are measured by Jaccard, Bleu, Sumo,

Embedding, and the FPR/TPR of all possible discrimination thresholds are plotted on a ROC curve. It is visualized as shown in Figure 3.3.

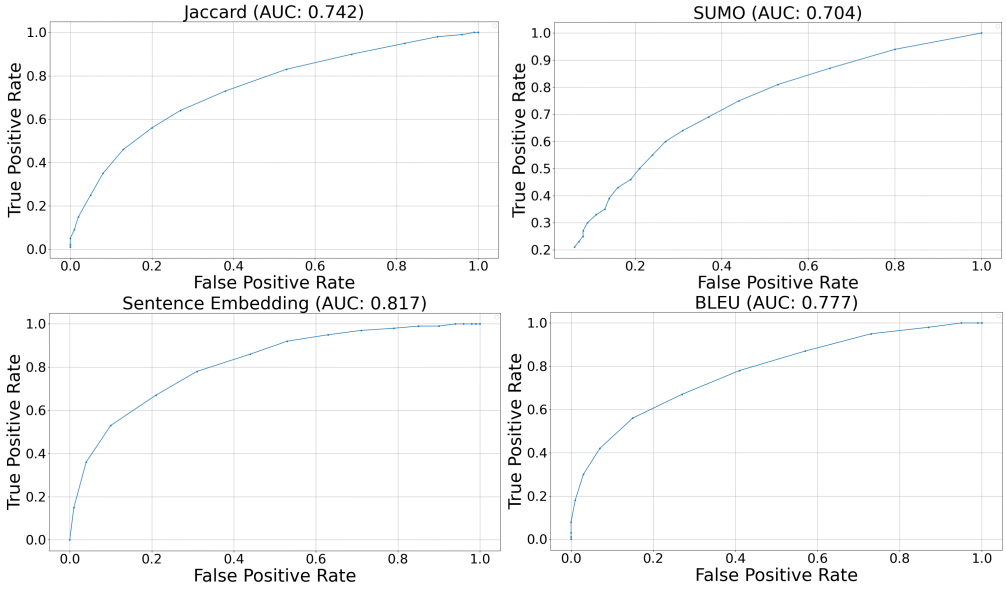

Figure 3.3: ROC curves with AUC of 4 metrics: Jaccard Similarity (upper left), SUMO (upper right), Sentence Embedding (lower left), BLEU(lower right).

In Figure 3.3, the performance of Sentence Embedding (0.817) appears the best among 4 metrics though the difference is not significant, followed by BLEU (0.777), Jaccard (0.742) and SUMO (0.704).

Moreover, instead of applying them independently, it is possible to consider some of their combinations as follows:

Figure 3.4 shows the result of experiments with combining the metrics pairwise. Each sub-figure shows a grid of scatter plots of metric pairs with density plots for individual metrics along the main diagonal. The two classes "paraphrase" (1) and "no paraphrase" (0) are represented by different colors. As can be seen from the large overlap of the class densities, no single metric distinguishes the classes well by itself. However, some metric pairs, e.g. BLEU and sentence embedding, appear to separate the classes better, as is apparent from the clustering tendency in the associated scatter plots, while the combination between BLEU & SUMO metric seems to be not a good classifier.

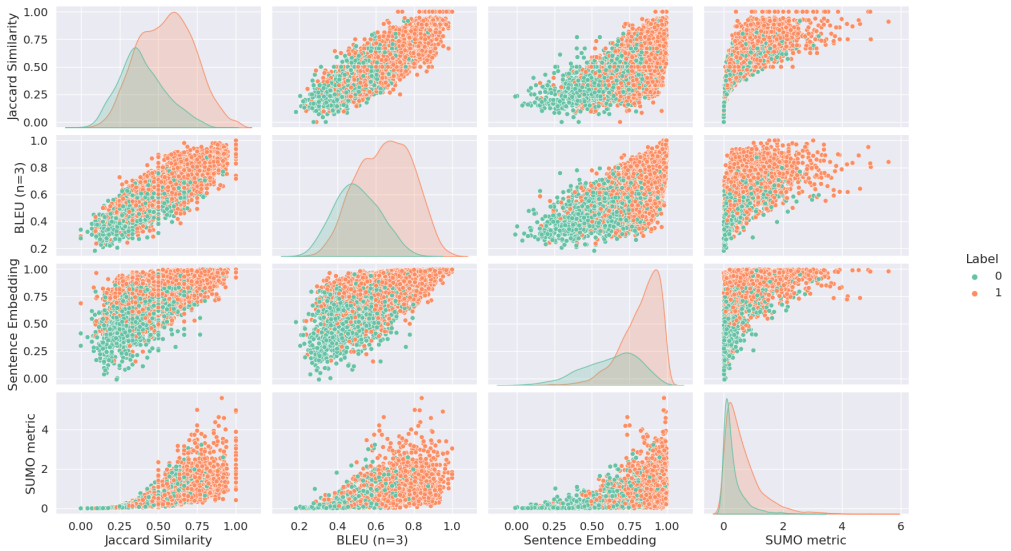

Figure 3.4: All possible combinations.

Based on scatter plot matrices in Figure 3.4, it is possible to see that applying a combination of metrics can produce a slightly better result than that in applying using metrics individually, even though there is no way to separate positive pairs (red color) and negative pairs (green color) perfectly. Specifically, in some combinations such as Sentence embedding and BLEU, the overlapping area is quite large. We can conclude that applying each metric individually or in a pair is not very efficient in distinguishing between positive and negative pairs. Hence, instead of using metrics separately or in a pair, we could use all of them as 4 features for Machine Learning algorithms.

3.2.4.4 Step 4: Classification Models for Paraphrase Detection.

In order to find an optimal combination of the four metrics, we train a set of machine learning models that predict the class "paraphrase" or "no paraphrase" using the metrics as features. 6 different Machine Learning algorithms are used in this project. They are Logistic Regression, K-nearest neighbors, Support vector machines, Random Forest, Naive Bayes, Gradient Boosting.

In this step, 4 metrics (Jaccard Similarity, BLEU, SUMO, Sentence Embedding) are used as features, while 6 Machine Learning algorithm are used as classifiers to discriminate between positive pair (paraphrasing) & negative pair (not-paraphrasing).

When applying all 4 features into lists of potential paragraph pairs, a table is generated (Figure 3.5):

ID	Text 1	Text 2	Jaccard	Sent_Trans	SUMO	BERT	Label
10635	Protesters holding ...	Protesters wear ...	0.409	0.777	0.020	0.891	**1**
19429	Hennepin County ...	Hennepin County ...	0.394	0.750	0.033	0.881	**1**
5198	In this ...	Chief James ...	0.230	0.603	0.008	0.842	**0**

Figure 3.5: Results when applying 4 metrics into potential paragraph pairs. They are the training data for machine learning algorithms. The last column is the class assigned by annotators: 1 is "paraphrase" and 0 is "no paraphrase"

The first column is the ID of each potential paragraph pair. The next 2 columns are their contents. The most important thing from the last 4 columns is the score from applying each metric/method into each pair.

3.2.5 Active Learning

After the end of Step 3, we already have a list of over 35 thousands potential paraphrase pairs, together with their corresponding values of four features i.e. Jaccard Similarity, BLEU, SUMO, Sentence Embedding.

However, all of them are still unlabeled data, because we do not know whether they are actually paraphrased or not. Assigning labels for all of them (35 thousand pairs) is a time and cost-intensive task. We need to find the way to have enough data for the very last step of this project which is to train a deep learning model of Paraphrase Generation and always requires a huge amount of labeled data.

Traditionally, it is necessary to gather a large amount of data randomly sampled from the underlying distribution and use them to train a model that can perform some prediction. This process is called passive learning. Notably, one of the biggest limitations of passive learning is that it takes a huge amount of time to collect labeled data. It might not be problematic to manually assign labels when there are several samples. Yet, it would be extremely time consuming to do the same for approximately 35,000 potential paraphrased pairs.

It is therefore critical to use Active Learning. There are situations in which unlabeled data is abundant but manually labeling is expensive and time-

consuming. Active learning algorithms can quickly query the user/teacher for labels. Active learning is a case of semi-supervised machine learning (Datacamp [2018]).

Active Learning (AL) aims to reduce the amount of data annotated by human experts. It is an iterative cyclic process between a teacher (usually the human annotator) and an active learner. In contrast to the data which is simply fed to the algorithm randomly in passive learning, the active learner can choose which samples are to be labeled next in an active learning process. When receiving new labeled data, the active learner trains a new model and the process starts from the beginning again.

In this project, active learning is applied as illustrated in Figure 3.6:

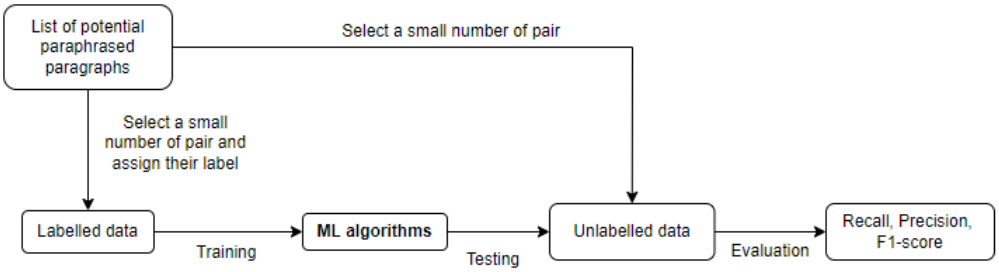

Figure 3.6: The details of each iteration in the Active Learning process

We randomly sample 100 examples to be labeled as part of the active learning step. Their respective label is 1 if they are paraphrased to each other, otherwise it is 0.

There are 2 annotators to work in this step. The rule of agreement in this step is that a label is assigned for a sample only when it receives the same opinion from both annotators, otherwise they are skipped.

Selected criteria are applied for defining the label of each sample. While there are many ways to identify paraphrasing, Mccarthy et al. [2009] suggest 4 dimensions to be used, namely Semantic completeness, Lexical similarity, Syntactic similarity, Paraphrase quality. Accordingly, in this step, if a pair of paragraphs cover almost all of crucial information such as the main object, event and the same time or date, even when the sentence structures are not identical, they are assigned as paraphrased, otherwise they are not paraphrased.

6 ML algorithms are trained independently by using labeled data in the previous step.

These trained ML algorithms are used to automatically assign labels for the remaining data to get these classifier confidence values (It is a pair of 2 float values for the probability of belonging to each class).

For example, if the confidence score of a paragraph pair is [0.3 0.7], it means that it is 30% likely that they are not paraphrased and 70% that they are paraphrased to each other.

In order to train these ML algorithms in the next iteration, a small number of predicted samples should be chosen to assign labels manually. In this step, they are chosen via the combination of majority voting & confidence score.

Majority Voting: When 6 machine learning algorithms are used to predict labels for a pair of paragraphs, if at least 4 votes are positive labels, the predicted label for this pair is paraphrasing, otherwise it is not. Majority Voting helps us to have more reliable results, when the decision is based on multiple classifiers instead of only a single classifier.

Confidence Score: We decided to choose a large number of samples with relatively high confidence values (e.g: more than 0.6), a small amount of samples with confidence value close to 0.5, and a small amount of samples with low confidence value. The reason behind this is that we want to especially avoid wrong "is a paraphrase" classifications with high confidence, because we wish to reach a high precision in the paraphrases we identify. So we double-check many of the high-confidence predictions. Predictions around 0.5 are interesting because they might contain patterns that the previous training set didn't contain and could help improve the classifier a lot. Low confidence predictions, i.e. high confidence for "not a paraphrase" are less interesting. This is because, for the subsequent paraphrase mining task, precision is more important than recall. Meanwhile we want an accurate dataset first and foremost, the number of samples it contains is secondary to that. In other words, the quality of this dataset is more important than its quantity.

Simultaneously, a confusion matrix is created based on the comparison amongst predicted labels from machine learning algorithms and ground truth of annotators.

The process is repeated for multiple times and stopped if and only if the performance of the model is not improved in comparison to the last iteration. This could be monitored and evaluated via a confusion matrix, F1 score, and recall on each iteration. In each iteration, the precision value is compared to itself in the previous iteration to see whether it is improved or not. The accuracy can

also be considered as well to have better observation. The training is stopped when there is no improvement on these values.

Finally, when this trained model is used for assigning labels for the entire list of 35,000 potential paraphrase pairs, there are approximately 6000 positive pairs (paraphrasing), which can be used as a training data for the deep learning model in subsequent steps.

3.3 Evaluation

The quality of this process is evaluated repeatedly via each iteration of the active learning process in Table 3.9. It is possible to see that, during the training process, the performance of the model improves over each and every iteration. However, after the fifth iteration, its performance starts to get worse, which can be interpreted as an overfitting signal. Therefore, the training is stopped at this point in order not to waste time and reduce training efficiency.

Table 3.9: Performance of classifiers in Active Learning process. (In each iteration, this model is evaluated by 300 samples)

Iteration	Num of training sample	Precision	Recall	F1 Score	Accuracy
1	100	0.9159	0.4033	0.5600	0.4867
2	200	0.9492	0.5545	0.7000	0.6800
3	300	0.9462	0.5591	0.7029	0.6533
4	400	0.9627	0.5945	0.7350	0.6900
5	500	**0.9856**	**0.6171**	**0.7590**	**0.7100**
6	600	0.9837	0.5193	0.6798	0.6200
7	700	0.9769	0.5359	0.6921	0.6233

Chapter 4

Paraphrase Generation

After being collected, approximately 6000 pairs of paraphrased paragraphs are collected, they are used as training & testing data for a text generator, which is used to write paraphrased texts with the given input text.

4.1 Models

A deep learning model for paraphrase generation is trained to automatically write a paraphrased text, given an input text.

4.1.1 Introduction

Once the above-mentioned steps as described in Chapter 3 are completed, a labeled dataset is available. Then, it is possible to start training a sequence to sequence model and generate paragraph-level paraphrases. The deep learning model used in this process is the summarised model of Huggingface Transformers library.

The rationale for this choice is that both summarization & paraphrasing share many similarities. However, one of the key differences lies in the length when summarization tends to be shorter than the original text while paraphrasing often results in similar lengths. Therefore, in using this model, I have changed a number of parameters in the training process. More specifically, the setup

for the maximum length of the output text is similar to that of the input. This helps make the output text similar to the output in terms of the length, which is in line with paraphrasing.

Huggingface Transformers is first introduced in Guyon et al. [2017] as a list of many pre-trained models. It has a variety of applications such as text classification, summarization, etc...

In this project, it is a deep learning model that uses a pair of encoder and decoder, then jointly learn them to convert input texts (sentences, paragraphs, document) into the same level of output texts, which have a different structure but convey the same meaning. Its general architecture is visualized in Figure 4.1 and Vaswani et al. [2017].

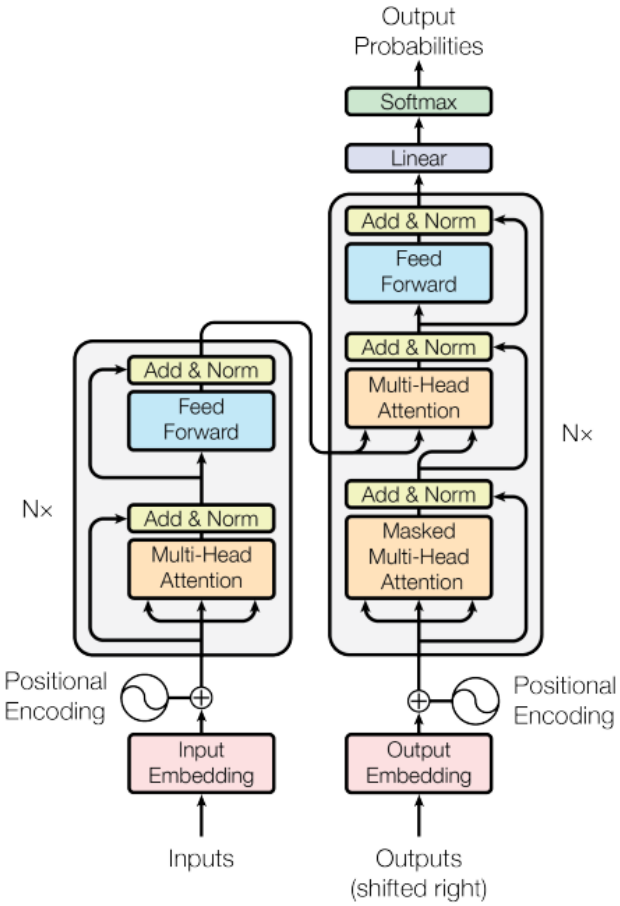

Figure 4.1: The Transformer - model architecture (Guyon et al. [2017])

As shown in Figure 4.1, there are 2 main components in this model of summarization: Encoder & Decoder. The first component of the summarised model used in this thesis, **Encoder**, is used to convert input text (e.g: paragraph) into an intermediate format (in this case, a numerical vector) and then, the second component, **Decoder**, converts it back into human-understandable format (in this case, a paraphrased text of input text). This model was trained on several datasets such as Workshop on Statistical Machine Translation (WMT), International Workshop on Spoken Language Translation (IWSLT), CNN/Daily Mail, The Extreme Summarization (XSum). (Wolf et al. [2020])

4.1.2 How to work the deep learning model

In general, the operation of this model can be divided into six following steps:

Step 1: The input text is put into Embeddings (with Position Encoding) and fed to the Encoder.

Step 2: The stack of Encoder processes it and returns an encoded representation of the input text.

Step 3: The target text is added with a start-of-text token, then transformed into Embeddings and fed to the Decoder.

Step 4: The stack of Decoders processes this along with the Encoder-stack-encoded representation to produce an encoded representation of the target sequence.

Step 5: The output layer converts it into word probability as well as generated output text which are paraphrased to input texts.

Step 6: The transformers model calculates loss value by comparing between generated text and target text from the training data. This loss is used to calculate gradients to train Transformers model during back-propagation step.

4.1.3 How to train the model

The collected data from Chapter 3 is split into a training set & a validation set with a reasonable ratio which is 90% to 10% (5400 examples and 600 examples for training step and testing step, respectively) in this case. The structure of

the training & validation files is shown in Figure 4.2. The output of this step is a fine-tuning model, based on the pre-trained model of DistilBART.

Source	Reference
Ron Rockwell Hansen, 58, of Syracuse, is a former Defense Intelligence Agency officer who was arrested Saturday at Seattle-Tacoma International Airport attempting to board a flight to China. The U.S. Attorney's Office for Utah accuses him of attempted espionage — receiving hundreds of thousands of dollars for trying to transmit defense information to the People's Republic of China's Intelligence Service.	Ron Rockwell Hansen, 58, a resident of Syracuse, Utah, and a former Defense Intelligence Agency officer, was arrested Saturday afternoon on federal charges including the attempted transmission of national defense information to the People's Republic of China. The FBI agents took Hansen into custody while he was on his way to Seattle-Tacoma International Airport in Seattle to board a connecting flight to China.
Since Obama has been unable to persuade the U.S. Congress to lift a longstanding trade embargo citizens are still prohibited from visiting Cuba as tourists. The U.S. government has approved exceptions to the ban, ranging from cultural, religious and educational travel to business and visiting family.	U.S. citizens are still prohibited from visiting as tourists, although there have long been exceptions to the ban, ranging from visiting family to business, cultural, religious and educational travel. The Obama administration has further eased the restrictions.

Figure 4.2: Training & Validation sample.

We choose the DistilBART model , which is a distilled version of BART (Bidirectional and Auto-Regressive Transformer)(Lewis et al. [2019a]). In this paper, BART is proved that particularly effective when fine tuned for text generation. Meanwhile the DistilBART model is a smaller, faster, cheaper and lighter version of the original BART model, so it helps us achieve both targets, namely good performance model & a reasonable running time model.

For the configuration of this DistilBART with fine-tuning model, the batch size parameter controls the number of training samples to work through before internal parameters of the model are updated. We set its value to 2 because of two main reasons: Firstly, there are only approximately 5500 samples in our training data. Therefore, it is not necessary to setup big batch sizes to reduce computation time. Secondly, larger batch sizes will lead to poorer generalisation. For the number of epochs in this training process, we set its value to 5. Actually, it is difficult to identify how many epochs could be enough for this process. Too few epochs will lead to insufficient training time for the model to learn how to paraphrase text effectively. On the other hand, if there are too many epochs in a small dataset in the training step, it can become overfitting, given the huge transformer models. Therefore, I set up the number of epochs to be 5, with the expectation that it is possible to balance the above two factors.

For remaining parameters, default values from original source code are used:

- Learning rate: 1e-5;
- Loss function: The combination between Cross-entropy (a loss function) & Label Smoothing (a regularisation technique).
- Optimization: Adam

After the training process is completed, a model weight file is created and it can be used in future to automatically generate paraphrased texts.

4.1.4 How to use the trained model

There are two ways for the trained model to use. In the first option, the trained model is loaded directly and then used to generate text, given an input text. In other words, whenever we want to use this trained model, we just need to load it by using built-in functions in popular deep learning frameworks such as Pytorch, Keras, Tensorflow. In the second option, to facilitate the manipulation of this model, particularly for those who do not know programming to use this model easily, I have developed a simple software by using Tkinter, a standard Python interface to the Tcl/Tk GUI toolkit. It can help users to use directly more easily and effectively. Accordingly, users can easily use this model as follows:

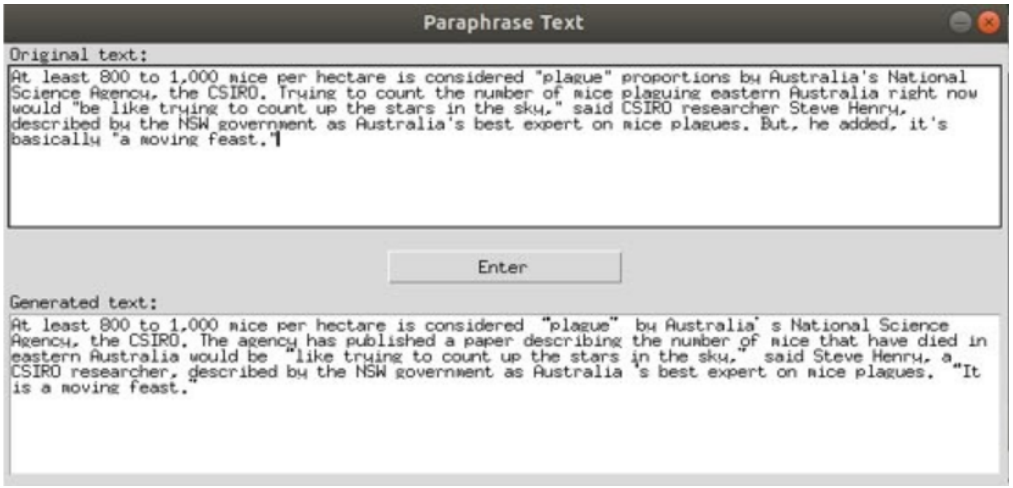

Figure 4.3: The interface of the paraphrasing tool

Figure 4.3 shows the application window with the developed user interface. In the text box in the upper half of the window, the user can enter any text of

their interest, and then click the central button. This starts the paraphrase generation, using the model trained in the previous section, with the result shown in the lower half of the window.

As a result, it is possible to see the generated text created successfully in the lower area of this tool.

4.2 Evaluation

The performance of the trained model is evaluated with quantitative & qualitative methods which are particularly covered in Section 4.2.3 and 4.2.4, respectively.

4.2.1 Dataset

As inputs for the evaluation process, original texts include 50 paragraphs collected randomly from 2 websites, namely CNN News & Wikipedia, while outputs are texts generated by the trained model

4.2.2 Model

There are 3 models used in this experiment, which are a fine-tuned model (Distilbart-with-FT) and 2 baseline models (Distilbart, Pegasus).

Distilbart model: DistilBART is a Natural Language Processing (NLP) Model which is implemented in the Transformers library, generally with the Python programming language. It is based on Facebook BART model, a sequence-to-sequence architecture combining a BERT (Devlin et al. [2018]) like bidirectional transformer encoder with a GPT (Improving Language Understanding by Generative Pre-Training) like a transformer decoder. It was trained on the basis of a combination of books and Wikipedia data. (Lewis et al. [2019b])

Pegasus model: It was first introduced in Zhang et al. [2019a]. This model uses self-supervised objective Gap Sentences Generation (GSG) to train a transformer encoder-decoder model. It was trained on C4 (the Colossal and

Cleaned version of Common Crawl), which consists of text from Web-pages and HugeNews, a dataset of articles.

Key distinctions between Baseline & Fine-Tuned Models are:

A baseline model just involves loading a pre-trained model and then uses them for direct testing with a test set. With the baseline model, it is very simple and fast. This is because, after models are loaded, it is possible to use them immediately without any further steps. However, the performance of the baseline model is usually not so good, especially when the difference of characteristics between pre-trained data and test data is significant.

On the other hand, fine-tuning is a way of applying or utilizing transfer learning. Specifically, fine-tuning is a process that takes a model already trained for one given task and then tunes or tweaks the model to make it perform a second similar task. After the Distilbart is loaded as a baseline model, it is continuously trained by the collected paragraph-level paraphrase dataset, which is mentioned in Chapter 3. Parameters for this training process such as batch size, learning rate, are also mentioned in Section 4.1.3.

4.2.3 Quantitative Evaluation

A set of metrics is used for evaluating the quality of text which is generated by the trained model.

4.2.3.1 Metrics

To automatically evaluate the similarity between source texts & generated texts in this project, ROUGE, or Recall-Oriented Understudy for Gisting Evaluation is used (Lin [2004]). It is a set of metrics used for comparing an automatically produced text against a source text.

In this experiment, 3 metrics of evaluation are applied:

Rouge-1: Overlap of unigram (each single word) between the source text and the generated text.

Rouge-2: Overlap of bigram between the source text and the generated text.

Rouge-L: Longest Common Subsequence (LCS) based statistics. The longest common subsequence problem takes into account natural sentence-level structure similarity and identifies the longest co-occurring in sequence n-grams automatically.

4.2.3.2 Results

In each of the metrics as mentioned in Section 4.2.3.1, there are 3 values for recall, precision and F1-score. The results are shown in Table 4.1. It shows means and standard deviations for recall, precision and f1-score (inner rows) for each of the Rouge-1, Rouge-2, and Rouge-L metrics (column groups) across three different paraphrase generation models (Distilbart with and without fine-tuning and Pegasus). The results reveal that fine-tuning appears to degrade the paraphrase generation performance compared to the pretrained Distilbart model, which outperforms Pegasus by a wide margin. There are two main reasons for such results.

	Rouge-1		Rouge-2		Rouge-L	
	Mean	Std	Mean	Std	Mean	Std
Distilbart-with-FT						
Recall	0.82	0.13	0.67	0.16	0.79	0.13
Precision	0.70	0.14	0.58	0.17	0.68	0.15
F1-score	0.75	0.13	0.62	0.16	0.73	0.13
Distilbart						
Recall	0.92	0.07	0.83	0.12	0.91	0.08
Precision	0.73	0.13	0.65	0.15	0.72	0.14
F1-score	0.81	0.09	0.72	0.12	0.80	0.10
Pegasus						
Recall	0.81	0.09	0.65	0.14	0.73	0.14
Precision	0.65	0.12	0.51	0.15	0.58	0.15
F1-score	0.72	0.10	0.57	0.15	0.64	0.14

Table 4.1: Recall, Precision & F1-score when using ROUGE metrics to evaluate the quality of all 3 models.

Firstly, it is to do with the comparison between the two versions of Distilbart models. The Distilbart without fine-tuning was originally designed for

summarization tasks. Therefore, it tends to keep/preserve key and important contents of the paragraph. That is the reason why the majority of the contents of the generated text is similar, or even identical, to the source text. In contrast, the Distilbart with fine-tuning has been trained with a paragraph-related dataset. Therefore, it tends to change all the used words and diversify the sentence structure in the paragraph. As a result, the contents of the generated text can have more changes as compared with the source text. The metric applied in this case, ROUGE, calculates the performance of the model based on the overlap of unigram, bigrams, etc. In this line, it is clear that, based on ROUGE metrics, Distilbart without fine-tuning will lead to better results than Distilbart with fine-tuning.

Secondly, the performance of two Distilbart models (with & without fine-tuning) is better than that of the Pegasus model. This might be due to the training set that these models use. According to the original papers of these models by Lewis et al. [2019b] and Zhang et al. [2019a], while Distilbart models are trained by a combination of books and Wikipedia data, the Pegasus model is trained with C4 (the Colossal and Cleaned version of Common Crawl), which consists of texts from Web-pages and HugeNews, a dataset of articles. Meanwhile, the text set used for both two models, as already mentioned in Section 4.2.1, consists of paragraphs collected randomly from two websites of CNN News and Wikipedia. Therefore, to a certain extent, Distilbart is more advantageous when its training set includes data from Wikipedia while that of the Pegasus model does not. This might be one of the reasons for the performance of Distilbart modes to be better than that of Pegasus model with ROUGE metric.

In order to test this hypothesis, we also compare how humans evaluate the generated paraphrases in the following section.

4.2.4 Qualitative Evaluation

5 annotators are applied to answer 6 questions for each pair of paragraphs as explained below.

4.2.4.1 Process

The application used in this crowdsourcing experiment is Label Studio (link). It is an open source tool of data which labels and explores multiple types of

data. It can be used for many types of data, from video, image to audio, texts. Furthermore, it can also be integrated with machine learning models to assign labels or perform iterative active learning.

For the crowdsourcing task in this project, 5 annotators are asked to use Label Studio to answer questions for a set of paragraph pairs (Figure 4.4).

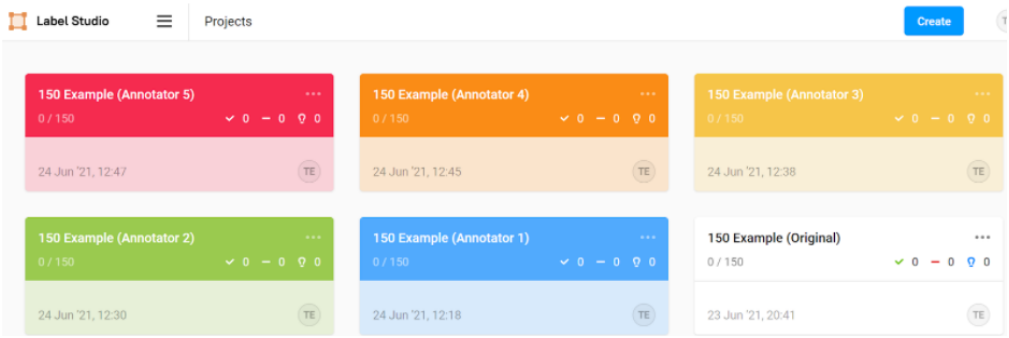

Figure 4.4: Each rectangle area is the task of annotation for an annotator

There are 6 questions that annotators should answer for each pair of paragraphs. They are

Question 1: Has the order of the sentences changed between the original text and the paraphrase? *(It checks whether the order of sentences is changed between the two paragraphs.)*

Question 2: Does the paraphrased text capture all crucial information in the original text? *(The paraphrase is semantically complete in relation to the source text. It conveys the same meaning as the source or at least the most important information.)*

Question 3: Does the paraphrased text contain information which does not appear in the original text? *(The paraphrase has information that is not found in the source text.)*

Question 4: Does the paraphrased text contain **incorrect** information which does not appear in the original text? *(The paraphrase wrongly modifies information from the source text.)*

Question 5: Does the paraphrased text contain **correct** information which does not appear in the original text? *(The paraphrase contains information that is not found in the source text, yet is correctly used (e.g. function words,*

synonyms, and modifiers that fit the context). In case of new facts such as names, locations, dates, events, annotators are allowed to verify such information via web search.)

Question 6: How do you rate the quality of the paraphrase? *(A good paraphrase should convey the same meaning as the original sentence, while being as different as possible on the surface form and being fluent and grammatical English.)*

In this experiment, with the combination between 50 paragraph pairs and 3 models (2 baseline and 1 fine-tuned models), there are totally 150 pairs of paragraph to be evaluated.

Each annotator is requested to answer 6 questions for each of 150 pairs. The result is saved into a .csv file as illustrated in Figure 4.5.

Source	Generated Text	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6
Published by the	The letter says tl	No	No	No	No	No	Very Poor
The houses are	The houses are	No	No	No	No	No	Poor
According to Em	1918 influenza	No	No	No	No	No	Acceptable
The party numbe	More than 200 n	No	No	No	No	No	Acceptable
Approximately a	Chauvin is charg	No	No	Yes	Yes	No	Very Poor
That's just part c	That's just part c	No	No	No	No	No	Acceptable
"We are all feelir	"We are all feeli	No	No	Yes	I don't know	I don't know	Acceptable
The suburb is a	The suburb is a	No	Yes	Yes	No	Yes	Poor
One Japanese a	Paralympic lege	No	No	Yes	Yes	No	Very Poor

Figure 4.5: Answers from 5 annotators.

Then, the results by all of the five annotators are consolidated as exemplified in Figure 4.6.

ID	Source	Generated Text	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6
0	Vito used to work long hours peeling sacks of garlic, making about $2 a day, while her husband worked as a day laborer in construction. But now their work has dried up, a casualty of an economic downturn in the Philippines after multiple coronavirus lockdowns. And trying to feed so many mouths has become a daily struggle for survival.	Vito and his wife, Silvio, used to work long hours peeling sacks of garlic, making about $2 a day. But their work has dried up, a casualty of an economic downturn in the Philippines after multiple coronavirus lockdowns. And trying to feed so many mouths has become a daily struggle for survival.	Yes: 0, No: 5	Yes: 3, No: 2	Yes: 4, No: 1	Yes: 3, No: 1, I don't know: 1	Yes: 0, No: 4, I don't know: 1	Very Good: 0, Good: 0, Acceptable: 1, Poor: 2, Very Poor: 2
1	Prime Minister Suga announced on February 9 that all foreign nationals would be barred from entering the country, but did not specify a start date or give details about how this would be implemented. The Tokyo Olympics are still scheduled to take place in July despite protests.	Prime Minister Yoshihide Suga announced on Feb. 9 that all foreign nationals would be barred from entering the country, but did not specify a start date or give details about how this would be implemented. The Tokyo Olympics are still scheduled to take place in	Yes: 0, No: 5	Yes: 5, No: 0	Yes: 4, No: 1	Yes: 0, No: 5, I don't know: 0	Yes: 4, No: 1, I don't know: 0	Very Good: 0, Good: 0, Acceptable: 1, Poor: 2, Very Poor: 2

Figure 4.6: The summarization for annotators' answer.

4.2.4.2 Results

In this experiment, there are 3 models, and for each model, there are 50 samples and 5 annotators. To put it differently, there are totally 250 answers for each question of each model. The final result is synthesized in Table 4.2 and Figure 4.7.

Table 4.2: Questions & Answers of manual evaluation process (Table)

Statistics of the manual evaluation from 5 annotators				
	Answer	Distilbart-with-FT	Distilbart	Pegasus
1. Order of sentences changed?	Yes	71	70	24
	No	179	180	226
2. All crucial information?	Yes	144	129	116
	No	106	121	134
3. Additional information?	Yes	90	38	18
	No	160	212	232
4. Additional **incorrect** information?	Yes	54	19	8
	No	31	15	8
	I don't know	5	4	2
5. Additional **correct** information?	Yes	15	14	1
	No	62	20	14
	I don't know	13	4	3
6. Quality of paraphrasing?	Very good	19	12	16
	Good	33	36	43
	Acceptable	80	68	98
	Poor	71	95	75
	Very poor	47	39	18

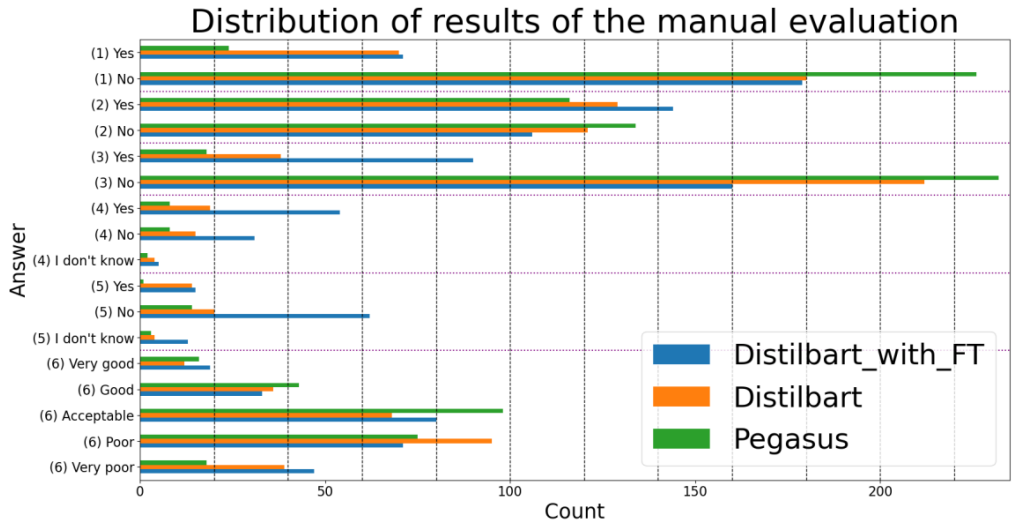

Figure 4.7: Distribution of answers to the human evaluation questions: (1) Order of sentences changed?, (2) All crucial information?, (3) Additional information ? (4) Additional incorrect information ? (5) Additional correct information ? (6) Qualities of paraphrasing ?

Question 1: Has the order of the sentences changed between original text and paraphrase ?

Notably, Figure 4.7 and Table 4.2 show that the number of answers "No" dominates that of "Yes". It means that the structure of generated texts is usually not different from the original texts among 3 models.

Question 2: Does the paraphrased text capture all crucial information in the original text ?

Figure 4.7 and Table 4.2 reveal that the fine-tuning model is generally better than baseline models in terms of keeping important information in original texts.

Question 3: Does the paraphrased text contain information which does not appear in the original text ?

Figure 4.7 and Table 4.2 show that the fine-tuning model generally often creates more new information which is not available in the original text than baseline models. To a certain extent, this proves the effectiveness of Active Learning.

Notably for **Question 4** & **Question 5**, when we evaluate whether gener-

ated text contains correct and incorrect information, which does not appear in source text. They are strictly related to Question 3. This is because, if the answer for Question 3 is No, it means that generated text has no new/additional information and we have no reason to ask question 4 & 5.

Table 4.2 illustrates how many times that annotators say **Yes** for Question 3 of each model

Based on this table, we can see that fine-tuning models usually contain more additional information than baseline models (90 versus 38, 18, respectively).

Question 4: Does the paraphrased text contain **incorrect** information which does not appear in the original text?

Question 5: Does the paraphrased text contain **correct** information which does not appear in the original text?

Based on Figure 4.7 and Table 4.2, we can see that, in all 3 models, whenever additional information appears in generated text, they usually are incorrect information. Moreover, the fine-tuning model often has a higher probability of containing incorrect information than in baseline models.

For Question 6 which is perhaps the most important question, we rank the quality of generated text in comparison with the source text. Many aspects are considered, including reserving crucial information, using different structures as well as words. We split them into 5 different ranks: Very good, good, acceptable, poor, very poor.

Question 6: How do you rate the quality of the paraphrase?

Figure 4.7 and Table 4.2 depict that the quality of "Acceptable" & "Bad" are the highest among all 3 models. It means that, from human perspectives, the quality of this model in paraphrasing text is not really good.

It is not a rigidly strict criterion to answer this question. Annotators can consider key points to have their reasonable answers as shown in Table 4.3:

Table 4.3: Evaluation for Question 6

Points to consider	**Ranking**
A pair of paragraphs is semantically the same and structurally grammatical diverse	Very Good / Good%
A pair of paragraphs is semantically the same and structurally similar but grammatically diverse and different in words	Acceptable / Good%
A pair of paragraphs is semantically different / lexicaly identical / ungrammatical	Bad / Very Bad%

To have a broader view of their quality, a simple solution is to group them into fewer levels of quality. An example is to group Very Good and Good into a group, Poor and Very Poor into a group. (Figure 4.8)

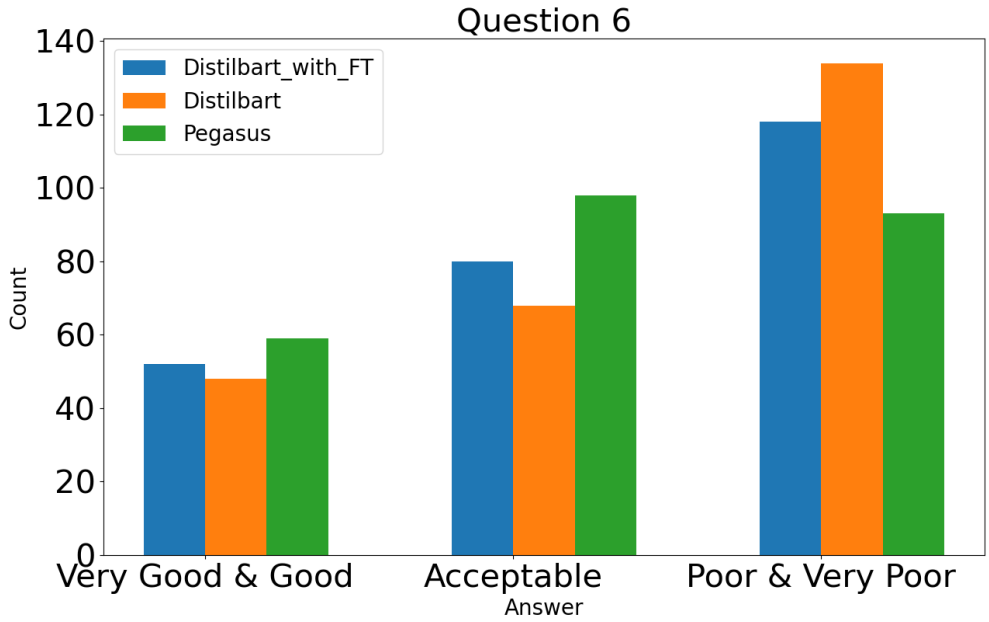

Figure 4.8: Answers for Question 6 (Summary quality with 3 levels).

Based on Figure 4.8, the number of Bad Quality is usually the highest in all 3 models (except Pegasus). Moreover, the number of Good Quality is always the lowest in all 3 models.

As shown in the Figure 4.8, the fine-tuning model (blue colour) increases both the number of Very Good and that of Very Poor paraphrases at the extremes, compared to two baseline models (orange & green colour). Possible reasons for

this include the training data used in the fine-tuning model. As already mentioned in Chapter 3, the fine-tuning model is trained by a dataset of paragraph-level paraphrases. Thus, It makes this model tend to change words as well as structure of the source text. Accordingly, in certain cases, it results in the generated text of the fine-tuning model to demonstrate better results than that of 2 baseline models which are pre-trained by the summarised dataset. In contrast, as fine-tuning model can only be trained with a considerably small number of paraphrase-related data (about 5400 samples), the performance is not stable, leading to the quality of generated text lower than that of 2 baseline models in many cases. Notably, while 2 baseline model are trained with a huge amount of data (about 160 gb data and 4500 gb data for Distilbart baseline model and Pegasus baseline model, respectively), the fine-tuning model is trained with only around 5400 samples, equivalent to 3 mb data.

Binning the quality levels into only three groups as illustrated in Figure 4.8 shows that fine-tuning improves the quality of the baseline DistilBART model rather consistently, but it does not reach the level of the Pegasus model. As already explained previously, a possible reason for this is that the Pegasus model has been trained with a huge amount of data, almost 30 times more than that done for DistilBart baseline & DistilBart with fine-tuning models. In general, the Pegasus model often demonstrates better than the other 2 models. However, it is possible to see that the number of "Very Good & good" annotators for thePegasus model is not too much higher than that of the Distilbart with fine-tuning model. Arguably, if the fine-tuning model can be trained with more data, its performance can be better than Pegasus.

Chapter 5

Conclusion

Based on available datasets, outcomes of previous research as well as studies on paraphrasing at the sentence level, this thesis develops a model which can paraphrase long texts with complex structure.

Additionally, we also labelled a dataset of pairs of paragraphs which are paraphrases of each other. It could be used as a reference for similar studies in the future. We had to manually label the dataset, since most of currently available datasets related to paraphrasing are in short-text levels such as sentences.

5.1 Achievement

2 main goals of my thesis have been completed:

Based on available datasets for summarization (WCEP & MultiNews), we collected and created a dataset containing approximately 6000 pairs of paragraphs by comparing and extracting paragraph in different articles. The dataset is used directly in this thesis, and could be used as a reference for future researches.

We trained a text generator. When a paragraph is provided as inputs, the generator will generate a text with similar contents but different structures as outputs. The main difference between this generator and other ones in previous researches is that the input is long and complicated paragraphs rather than single sentences. There are 3 model used and compared to one another

in this thesis: 2 base line models (DistilBart & Pegasus) and a fine-tuning model (based on DistilBart). They were automatically evaluated with ROUGE metrics by 5 annotators. Based on this evaluation, it is obvious that the fine-tuning model outperform the base line model.

5.2 Challenges

There are some challenges limiting the performance of the the generator

5.2.1 Paragraph-level focus

In this thesis, the main object is long texts, which has a more complicated structure than short texts on previous publications, specifically when we focus on paragraphs. Therefore, it is more difficult to research, evaluate as well as compare them.

5.2.2 Data availability

Moreover, there is a lack of labelled data. As mentioned on Chapter 1, almost all datasets for paraphrasing are in short-text levels. Hence, when carrying out research on paragraph levels, we have to deal with the shortage of suitable labelled data. Secondly, most metrics/methods which have been used for short texts reveal inefficiency when they are applied for long and complicated texts such as paragraphs.

Among other difficulties, one of the most important tasks in this thesis is to collect data. Specifically, we need to consider pairs of articles to find paragraphs which are paraphrases of each other. At sentence level, this task seems to be quite straightforward. However, at paragraph level, it is significantly more complicated since we have to deal with various scenarios as illustrated below.

Scenario 1: There are sentences in the reference article which are paraphrases of sentences in the source article. However, they are not in the same paragraph, which makes the matching process become harder and more time-consuming, and almost impossible when the volume of data is large.

Scenario 2: The main part of 2 paragraphs are somehow similar. However, the remaining part is either only mentioned in 1 paragraph, or totally different in 2 paragraphs. In this case, it is not straightforward for annotators to determine if the 2 paragraphs are paraphrases of each other or not, since it depends on many other factors.

Scenario 3: Content of paragraphs is more complicated than sentence's one. For example, 2 paragraphs could both describe the same event which happens in the same place at the same time. However, they focus on different aspects of the event. Hence, It is hard for annotators to determine whether they are paraphrases of each other or not.

To train a text generator effectively, it is crucial that we need sufficient data. However, since available datasets only focus on short texts, we have to collect data ourselves (Chapter 3). This implies the fact that we do not have enough dataset (6000 samples). As a result, the training process is not really as efficient as we expected.

5.2.3 Annotator inconsistency

Another challenge is about qualitative evaluation (Section 4.2.4) when 5 annotators are asked to rate the quality of the paraphrase. Although there are standards based on similarity of difference in semantic, structure or grammar defined so as to support them in performing this task, the discrepancy amongst annotators' evaluation is still significant. For example, there are many pairs of paragraphs which are rated at different levels, like very good, good and bad by annotators. As a result, it is not easy to collect annotators' consistent evaluation, leading to less reliable final evaluation.

5.2.4 Imperfect ways of text paraphrasing

There is still a far distance from what a human can do to paraphrase. Normally, there are several ways to paraphrase a text:

The simplest method is to **use similar words**: A word could be replaced by its synonym. We can also use a method of **substituting definitions**: Some key words in the original text may be replaced by their definitions in paraphrased text. **Switching the order of clauses** is a good option when a sentence is

rewritten by switching the order of the original clause. Another method is **to change the voice**, when active voice and passive voice is swapped to each other. The last three methods are: **Exchanging verbs & nouns**: Verbs in the original text are changed to their equivalent nouns in the paraphrase and vice versa, **Combining sentences**: Sentences from the original texts are combined by using conjunctions or relative clauses, and **Using creatively time, numbers and dates**: Different ways are applied to demonstrate numbers, statistics, dates and times (for example: 12 weeks = 3 months = 84 days).

In this thesis, our paraphrase generation models generate paraphrased text only by using similar words, combining sentences, and switching the order of clause, while other methods are almost never used. The main reason is that the performance of a deep learning model (here is a text generator) depends on the training data. In our case, articles are from 2 datasets (MultiNews and WCEP), which do not reveal diversity in writing style. This leads to the fact that there is no pair of paragraphs that use methods such as "Change the Voice" or "Creative use of Time, Numbers & Dates". Therefore, our text generator, which is trained by these data, has no way to paraphrase using these advanced methods.

5.2.5 Inadequately clear paraphrasing results

Moreover, paraphrasing results are not quite clear. Specifically, in many pairs of original text and generated text, many contents still overlap to each other. There are 2 main reasons:

The first reason: The quality of training data. Contents of pairs of paragraphs in training data overlap to each other quite often, affecting the performance of the text generator.

The second reason: The requirements is higher for text paraphrasing in compared to other common tasks in NLP like text classification or summarization. In other tasks, we only need to determine the topic of text (text classification) or machine translation. Nevertheless, in text paraphrasing, we need to not only select synonyms but also change the structure of texts while maintaining the contents of the source texts

5.3 Future Work

Through this thesis, we can identify a couple of aspects in previous researches for further considerations, includes:

Metrics such as SUMO which was introduced in the paper "A Metric for Paraphrase Detection" specifically created for Paraphrase Detection. They are proved for the efficiency in corpus like The Microsoft Paraphrase Corpus or The Knight and Marcu Corpus. However these 2 corpus are both at the sentence level. When they are applied on our dataset, the performance is not really good. There is no significant difference between the positive pairs and negative ones.

Jaccard similarity or BLEU have been widely used in previous researches relating to paraphrase. However, its performance is poor as well. Hence, we can conclude that, when objects are texts with high complexity, such as paragraphs as in this thesis, using only one single metric/method is not efficient enough. Instead, using multiple metrics/methods could bring about better results.

Moreover, we would like to make some recommendations for improvements as follows:

The quality of the text generator can be improved by training more data. To do this, more datasets from more different sources should be extracted, analysed instead of only two data sets used in this project.

There should be more researches and applications of multiple metrics/methods as well as other ML algorithms to enhance the quality of filtering potential paraphrasing data.

We would like to recommend usage of other models such as deep learning models, based on a combination of deep generative models (VAE) with sequence-to-sequence models (LSTM) (Gupta et al. [2018a]) or more intelligent models based on Deep Reinforcement Learning (Li et al. [2018]) to compare their performance and choose the best ones in order to consider fine-tuning steps.

Bibliography

Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 1415–1425. The Association for Computer Linguistics, 2014. doi: 10.3115/v1/p14-1133. URL https://doi.org/10.3115/v1/p14-1133.

Rahul Bhagat and Eduard H. Hovy. What is a paraphrase? *Comput. Linguistics*, 39(3):463–472, 2013. doi: 10.1162/COLI_a_00166. URL https://doi.org/10.1162/COLI_a_00166.

William Blacoe and Mirella Lapata. A comparison of vector-based representations for semantic composition. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 546–556. ACL, 2012. URL https://aclanthology.org/D12-1050/.

Andrei Z. Broder. Identifying and filtering near-duplicate documents. In Raffaele Giancarlo and David Sankoff, editors, *Combinatorial Pattern Matching, 11th Annual Symposium, CPM 2000, Montreal, Canada, June 21-23, 2000, Proceedings*, volume 1848 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 2000. doi: 10.1007/3-540-45123-4_1. URL https://doi.org/10.1007/3-540-45123-4_1.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. Joint copying and restricted generation for paraphrase. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3152–3158. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14527.

Joao Cordeiro, Gael Dias, and Pavel Brazdil. A metric for paraphrase detection. In *2007 International Multi-Conference on Computing in the Global Information Technology (ICCGI'07)*, pages 7–7, 2007. doi: 10.1109/ICCGI.2007.4.

Dipanjan Das and Noah A. Smith. Paraphrase identification as probabilistic quasi-synchronous recognition. In Keh-Yih Su, Jian Su, and Janyce Wiebe, editors, *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 468–476. The Association for Computer Linguistics, 2009. URL `https://aclanthology.org/P09-1053/`.

Datacamp. Active learning: Curious ai algorithms. `https://www.datacamp.com/community/tutorials/active-learning`, 2018. Accessed: 2018-02-19.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Kuntal Dey, Ritvik Shrivastava, and Saroj Kaushik. A paraphrase and semantic similarity detection system for user generated short-text content on microblogs. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2880–2890. ACL, 2016. URL `https://aclanthology.org/C16-1271/`.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing, 2005. URL `https://aclanthology.org/I05-5002/`.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers*, pages 1074–1084. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1102. URL `https://doi.org/10.18653/v1/p19-1102`.

Anthony Fader, Luke S. Zettlemoyer, and Oren Etzioni. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 1608–1618. The Association for Computer Linguistics, 2013. URL `https://aclanthology.org/P13-1158/`.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1156–1165. ACM, 2014. doi: 10.1145/2623330.2623677. URL `https://doi.org/10.1145/2623330.2623677`.

Andrew M. Finch, Young-Sook Hwang, and Eiichiro Sumita. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing, 2005. URL `https://aclanthology.org/I05-5003/`.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: the paraphrase database. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 758–764. The Association for Computational Linguistics, 2013. URL `https://aclanthology.org/N13-1092/`.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. A large-scale multi-document summarization dataset from the wikipedia current events portal. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1302–1308. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.120. URL `https://doi.org/10.18653/v1/2020.acl-main.120`.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. Search engine guided non-parametric neural machine translation. *CoRR*, abs/1705.07267, 2017. URL `http://arxiv.org/abs/1705.07267`.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. A deep generative framework for paraphrase generation. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5149–5156. AAAI Press, 2018a. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16353.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. A deep generative framework for paraphrase generation. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5149–5156. AAAI Press, 2018b. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16353.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Trans. Assoc. Comput. Linguistics*, 6:437–450, 2018. URL https://transacl.org/ojs/index.php/tacl/article/view/1296.

Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 2017. URL https://proceedings.neurips.cc/paper/2017.

Samer Hassan and Rada Mihalcea. Semantic relatedness using salient semantic analysis. In Wolfram Burgard and Dan Roth, editors, *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*. AAAI Press, 2011. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3616.

Aminul Islam and Diana Zaiu Inkpen. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Trans. Knowl. Discov. Data*, 2(2):10:1–10:25, 2008. doi: 10.1145/1376815.1376819. URL https://doi.org/10.1145/1376815.1376819.

Yangfeng Ji and Jacob Eisenstein. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 891–896. ACL, 2013. URL `https://aclanthology.org/D13-1090/`.

Tomoyuki Kajiwara. Negative lexically constrained decoding for paraphrase generation. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6047–6052. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1607. URL `https://doi.org/10.18653/v1/p19-1607`.

Zornitsa Kozareva and Andrés Montoyo. Paraphrase identification on the basis of supervised machine learning techniques. In Tapio Salakoski, Filip Ginter, Sampo Pyysalo, and Tapio Pahikkala, editors, *Advances in Natural Language Processing, 5th International Conference on NLP, FinTAL 2006, Turku, Finland, August 23-25, 2006, Proceedings*, volume 4139 of *Lecture Notes in Computer Science*, pages 524–533. Springer, 2006. doi: 10.1007/11816508_52. URL `https://doi.org/10.1007/11816508_52`.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019a. URL `http://arxiv.org/abs/1910.13461`.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019b. URL `http://arxiv.org/abs/1910.13461`.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. Paraphrase generation with deep reinforcement learning. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3865–3878. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1421. URL `https://doi.org/10.18653/v1/d18-1421`.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*, 2004.

Nitin Madnani and Bonnie J. Dorr. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Comput. Linguistics*, 36(3):341–387, 2010. doi: 10.1162/coli_a_00002. URL https://doi.org/10.1162/coli_a_00002.

Philip Mccarthy, Rebekah Guess, and Danielle McNamara. The components of paraphrase. *Behavior research methods*, 41:682–90, 09 2009. doi: 10.3758/BRM.41.3.682.

Rada Mihalcea, Courtney D. Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pages 775–780. AAAI Press, 2006. URL http://www.aaai.org/Library/AAAI/2006/aaai06-123.php.

Muhidin A. Mohamed and Mourad Oussalah. A hybrid approach for paraphrase identification based on knowledge-enriched semantic heuristics. *Lang. Resour. Evaluation*, 54(2):457–485, 2020. doi: 10.1007/s10579-019-09466-4. URL https://doi.org/10.1007/s10579-019-09466-4.

Bo Pang, Kevin Knight, and Daniel Marcu. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In Marti A. Hearst and Mari Ostendorf, editors, *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics, 2003. URL https://aclanthology.org/N03-1024/.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL, 2002. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek V. Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. Neural paraphrase generation with stacked residual LSTM networks. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on*

Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan, pages 2923–2934. ACL, 2016. URL https://aclanthology.org/C16-1275/.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. Paraphrase recognition via dissimilarity significance classification. In Dan Jurafsky and Éric Gaussier, editors, *EMNLP 2006, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, 22-23 July 2006, Sydney, Australia*, pages 18–26. ACL, 2006. URL https://aclanthology.org/W06-1603/.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1410. URL https://doi.org/10.18653/v1/D19-1410.

Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci. Natural language understanding with the quora question pairs dataset. *CoRR*, abs/1907.01041, 2019. URL http://arxiv.org/abs/1907.01041.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 801–809, 2011. URL https://proceedings.neurips.cc/paper/2011/hash/3335881e06d4d23091389226225e17c7-Abstract.html.

Alex Sokolov and Denis Filimonov. Neural machine translation for paraphrase generation. *CoRR*, abs/2006.14223, 2020. URL https://arxiv.org/abs/2006.14223.

Yiping Song, Rui Yan, Xiang Li, Dongyan Zhao, and Ming Zhang. Two are better than one: An ensemble of retrieval- and generation-based dialog systems. *CoRR*, abs/1610.07149, 2016. URL http://arxiv.org/abs/1610.07149.

Yu Su and Xifeng Yan. Cross-domain semantic parsing via paraphrasing. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of*

the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, pages 1235–1246. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1127. URL https://doi.org/10.18653/v1/d17-1127.

Brian Thompson and Matt Post. Paraphrase generation as zero-shot multilingual translation: Disentangling semantic similarity from lexical and syntactic diversity. In Loïc Barrault, Ondrej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno-Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation, WMT@EMNLP 2020, Online, November 19-20, 2020*, pages 561–570. Association for Computational Linguistics, 2020. URL https://aclanthology.org/2020.wmt-1.67/.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. Using dependency-based features to take the 'para-farce' out of paraphrase. In Lawrence Cavedon and Ingrid Zukerman, editors, *Proceedings of the Australasian Language Technology Workshop, ALTA 2006, Sydney, Australia, November 30-December 1, 2006*, pages 131–138. Australasian Language Technology Association, 2006. URL https://aclanthology.org/U06-1019/.

Su Wang, Rahul Gupta, Nancy Chang, and Jason Baldridge. A task in a suit and a tie: Paraphrase generation with semantic augmentation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7176–7183. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33017176. URL https://doi.org/10.1609/aaai.v33i01.33017176.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. In Nicoletta Calzolari, Yuji

Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 1340–1349. ACL, 2016. URL https://aclanthology.org/C16-1127/.

Wikipedia. Inflection — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Inflection&oldid=1055849076, 2022. [Online; accessed 08-January-2022].

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, RÃ©mi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. Response generation by context-aware prototype editing. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7281–7288. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33017281. URL https://doi.org/10.1609/aaai.v33i01.33017281.

Wei Xu, Chris Callison-Burch, and Bill Dolan. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (PIT). In Daniel M. Cer, David Jurgens, Preslav Nakov, and Torsten Zesch, editors, *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 1–11. The Association for Computer Linguistics, 2015. doi: 10.18653/v1/s15-2001. URL https://doi.org/10.18653/v1/s15-2001.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. *CoRR*, abs/1912.08777, 2019a. URL http://arxiv.org/abs/1912.08777.

Yuan Zhang, Jason Baldridge, and Luheng He. PAWS: paraphrase adversaries from word scrambling. In Jill Burstein, Christy Doran, and Thamar

Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1298–1308. Association for Computational Linguistics, 2019b. doi: 10.18653/v1/n19-1131. URL `https://doi.org/10.18653/v1/n19-1131`.