NON-UNIFORM LANGUAGE DETECTION IN TECHNICAL WRITING

by

Weibo Wang

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
April 2016

To my parents.

Table of Contents

List of Tables

List of Figures

Abstract

Technical writing in professional environments, such as user manual authoring for new products, is a task that requires uniform language and writing style to avoid ambiguity. However, products of modern industries tend to be sophisticated and thus, quite a lot technical description documents are created by co-authoring. Authors from different departments who expertize on different areas might be responsible for certain independent chapters. Considering that each author has different writing habit and some of them could make mistakes on use of terminologies, the integrated long document could be incoherent and ambiguous.

Non-uniform language detection aims to avoid such inconsistency for technical writings by detecting sentences in a same document that are intended to have the same meaning or usage within a similar context but use different words/writing style, and outputting them by pairs. This thesis proposes a solution that utilizes text similarity algorithms in lexical, syntactic, semantic and pragmatic levels. Some online resources such as WordNet, Flickr, and part-of-speech tagger from Stanford University are utilized. Different analysis results are integrated by applying a machine learning classification method. We tested our methods using smart phone user manuals and the results are evaluated in terms of recall ratio, precision, accuracy, and F-measure. Some up-to-date implementations in related area, such as paraphrase detection and near-duplicate document/text detection are reviewed. We compared these implementations with our solution by running the same data set. The experiments demonstrate that the proposed solution to NLD task is the most efficient method to date, and the final result is close to the upper bound performance.

List of Abbreviations Used

CNG Common N-gram.

ESA Explicit Semantic Analysis.

GTM Google Tri-gram Method.

KNN K-Nearest Neighbor.

LCS Longest Common Subsequence.

LSA Latent Semantic Analysis.

ND Near-duplicate Detection.

NDD Near-duplicate Document Detection.

NGD Normalized Google Distance.

NLD Non-uniform Language Detection.

NLDS Non-uniform Language Detecting System.

NLP Natural Language Processing.

NTD Near-duplicate Text Detection.

PD Paraphrase Detection.

POS Part-of-Speech.

RAE Recursive Autoencoder.

STS Semantic Text Similarity.

SVM Support Vector Machine.

UB Upper Bound.

Acknowledgements

First, I would like to thank my supervisor, Evangelos Milios. Thank you for taking me as a master student in the first place. Thank you for your guidance, patience. Thank you for assigning me the great project from the great cooperate company. You taught me how to read papers, how to discover and propose a real task, how to divide and conquer the problems, and how to organize my experiment results into academic format. I am so proud to do research with you.

I am also grateful to my mentors, Abidalrahman Mohammad and Aminul Islam. Thank you for sharing your experience and organize my thought when I was lost in so many ideas and tasks. You had many great ideas for solving hard problems which I could never think of before meeting you. Thank you for trusting me and always being patient and nice to me. I learned a lot from you.

I am fortunate to meet and work with wonderful people in Innovatia Inc. Thank you Andrew Albert, David Crowley, and Erika Allen for all of your time and efforts. You proposed valuable problems and trusted me, you spent time for me, and you provided me your invaluable data that I could never get anywhere else. Thank you for arrange great meetings which give me chances talking directly with the authors. That helped me a lot to define and address this NLD task. You always gave me valuable response and positive energy.

I would like to thank Dalhousie University to provide such a great environment to do research and allowing me to finish my work. Coming to Dalhousie was a great opportunity that has changed my life forever. It was my first time being away from home, and I am so enjoyed to study and do reasearch in such beautiful campus.

I am proudly grateful to my supportive and loving family. To my mom and dad, I am blessed to be your child. Thank you all for encouraging me to pursue my interests and your unconditional love and support. I am grateful to my friends who never let me be alone. Thank you for your companion and love.

Chapter 1

Introduction

Technical writing, such as creating device operation manuals and user guide handbooks, is a special writing task that requires accurate description to explain a certain product or operation. Considering that a technical document could be a product of co-authoring, the document could have inherit inconsistency in using of words, or have different writing style, which might be ambiguous to readers. To avoid such ambiguity and bring accurate and straightforward understanding to readers, technical writing always requires consistency in the use of terminology and uniform language. There are always demands from modern industries to improve the quality of technical documents in cost-efficient ways.

Non-uniform Language Detection (NLD) aims to avoid inner-inconsistency and ambiguity of technical content by utilizing text mining techniques to identify non-uniform sentences for authors. Such sentences are intended to have the same meaning or usage within a similar context but use different words/writing style. However, even though non-uniform sentences tend to have similar wording, similar sentence pairs do not always indicate a non-uniform language instance. For example, here are six pairs of similar sentence pairs cited from the iPhone user manual [3], where only three pairs are true non-uniform language instances:

```
if you don't see, tap the screen to show the controls.
if you don't see, tap the screen to display the controls.

Start writing the name of the item.
Start entering the name of the item.

dismiss the control menu without choosing an action.
dismiss the control menu without performing an action.
```

`if the photo hasn't been downloaded yet, tap the download notice first.`
`if the video hasn't been downloaded yet, tap the download notice first.`

`The home button is at the bottom center of the front of the device.`
`The home button is at the bottom center of the front of iPhone.`

`you can also turn blue tooth on or off in control center.`
`you can also turn wi-fi and blue tooth on or off in control center.`

As we can see above, the pattern of difference within each sentence pair could be between one word and one word, or one word and multiple words, or one sentence having extra words that the other sentence does not have. Each different pattern found could be a true or false non-uniform language instance, depending on the content and context.

For the first pair in the examples above, the word `show` and `display` are synonyms. Both sentences convey the same meaning so they are an instance of non-uniform language. For the second and the third pair, even though the words `enter` and `write`, `choose` and `perform` are not synonyms, since each sentence pair describes the same operation, they should be considered as non-uniform language as well. For the fourth pair, even though the only different parts between the sentences, `photo` and `video`, are both media contents, because they are different objects, they should not be detected as non-uniform language. For the fifth pair, the length of two sentences are not equal, and the word `iPhone` and the phrase `the device` is not related in either lexical level or semantic level. However, by checking the context, we do confirm that these two sentences are both intended to describe the home button of iPhone, so they should be considered as non-uniform language. And for the last pair, one sentence has an extra phrase `wi-fi and`, which introduced an extra operation. For this case, even a human judger could not tell whether it is non-uniform language or not without checking the context, since the

sentences might describe a same operation (non-uniform language), or two different operations (not non-uniform language). Therefore it is challenging to distinguish true and false occurrences of non-uniform cases based on text or word similarity algorithms only. More fine grained analysis needs to be applied.

NLD task is close to Paraphrase Identification or Paraphrase Detection (PD) area, which aims to detect sentences that have essentially the same meaning [14]. However, considering that paraphrase is a restatement using very different words, which aims to make it appear different from the original text, PD techniques could not perform well on the NLD task as they focus on different grained variations. As a matter of fact, we have reviewed studies in PD area, and found the most to date implementations of both supervised method [50] and unsupervised method [28], that related to the NLD task. However, all the four sentence pairs provided above would be recognized as paraphrases by these analyzers, even though only two of the pairs are real non-uniform language. Thus, state-of-the-art PD techniques are unable to make accurate judgment on these instances since PD could neither process nor analyze in sufficiently fine grain for the NLD task.

Another related area to NLD task is Near-duplicate Detection (ND). There are two subdomains of ND that worth mentioning: Near-duplicate Document Detection (NDD), and Near-duplicate Text Detection (NTD). NDD focuses on documents that are identical in terms of content but differ in a small portion of the document such as advertisements, counters and timestamps [38], and the documents that differ only slightly in content [47]. The near-duplicate documents are productions of web crawling and the automatic collecting function of digital libraries. For instance, the arXiv [55] allow users to submit academic papers for inclusion, and users might make minor revisions to a document and submit it as a new document rather than updating their existing submission. Similarly, CiteSeerX [43] automatically collect papers through focused crawling, similar versions of a paper may exist at multiple locations on the Web and these multiple versions may be automatically added to the collection as a result of the automatic crawling and ingesting. The near-duplicate documents are abundant on the Web. NDD aims to detect and eliminate such documents to save network bandwidth, reduces storage costs, and improves the quality of search indexes. Since NDD focuses on the

variations between two documents, especially the variations on metadata rather than the content, the NDD solutions could not fit well on NLD tasks.

NTD foucuses on short text such as mobile phone short messages, instant messages, or tweets, which are intended to have the same meaning but differ in terms of informal abbreviations, transliterations, and network languages [24]. The detection and elimination of near-duplicate text is of great importance for other text language processing, such as clustering, opinion mining, and topic detection [53]. However, the studies in NTD area focus on reducing the comparison time in large -scale text databases and creating informal abbreviation and expression corpus, rather than exploring the text similarity methods. Very basic similarity methods, such as "which returns high scores when two text strings share a large portion of identical or highly similar substrings. [53]", are utilized, and is obviously unable to solve NLD task.

This thesis proposes a solution for detecting non-uniform language within a technical document at the sentence level. Natural Language Processing (NLP) techniques in both lexical, syntactic, semantic, and pragmatic levels are utilized. The solution also integrates data sources such as Part-of-Speech (POS) tagging corpus [42], WordNet [40], Google Tri-gram Method (GTM) [6], and Flickr [19]. Analysis from different perspectives and NLP levels is applied and the results are regarded as independent features that are finally integrated by applying a classification method based on Support Vector Machine (SVM) to provide the final judgment on non-uniform language instances.

1.1 Research Objectives

The main goal of this research is providing a solution to help authors to improve the quality of technical writings by detecting non-uniform language within documents, so that the authors are able to fix the inconsistent parts in the document to make it less ambiguous and more accurate and straightforward to readers. In order to achieve our goal, we performed following studies:

1. Using the combination of text Google Tri-gram Method, Cosine similarity and Longest Common Subsequence (LCS) similarity to detect sentence that appear similar and related in both syntactic and semantic level.

2. Using part-of-speech tags to analyze similar sentence pairs and categorize the difference between each sentence pair into different patterns.

3. Using character N-gram and CNG distance [32] to measure the relatedness between the different parts of a similar sentence pair.

4. Using WordNet and Flickr to discover relationship such as synonym, metonymy between the different parts of a similar sentence pair.

5. Using word to word GTM score to measure the semantic relatedness between the different parts of a similar sentence pair.

1.2 Contributions

Here is the list of contributions of this thesis:

1. Propose the NLD task, which is novel in text mining area. Provide a novel solution that achieves good performance in terms of precision, recall ratio, accuracy, and F-measure. We defined the upper bound performance of this task by inviting three human experts to manually label similar sentence pairs as true or false non-uniform language instances, and evaluate their performance using Fleiss' Kappa analysis. The experiments demonstrate that the solution we proposed is close to the upper bound performance.

2. Review the studies in paraphrase detection area, found the state-of-the-art unsupervised PD system STS, as well as the state-of-the-art supervised PD system RAE, and utilize them as baseline methods of NLD task. We showed that state-of-the-art solutions are not able to provide sufficient fine grain analysis to detect non-uniform language for technical writing.

3. Review the studies in near-duplicate detection area, especially the two sub-domains: near-duplicate document detection, and near-duplicate text detection. We discussed the algorithms, focused target, as well as the limitations in this area. We showed the state-of-the-art solutions in such areas are not able to be used as the baseline methods of NLD task.

4. Propose and implement a general heuristic algorithm to determine the filtering thresholds for different text similarity methods.

5. Propose and implement a framework that integrates the similarity methods at lexical, syntactic, semantic, and pragmatic levels. We also showed different importance among different similarity methods by performing an ablation study.

1.3 Outline

Chapter 2 describes the current state in the field of paraphrase detection and near-duplicate document detection, and provides some background on the NLD topic. Chapter 3 discusses three text similarity methods that utilized to detect similar sentence pairs as non-uniform language candidates. Chapters 4 introduces the methods we utilized to distinguish true non-uniform language instances from false instances. Chapter 5 describes the experiments we performed to evaluate our method. Chapter 6 concludes this thesis.

Chapter 2

Background and Related Work

In this chapter, previous work in related fields is reviewed. To the best of our knowledge, the area of NLD is under-explored. Yet, there are many research studies performed in related areas such as PD and NDD. Some methods and implementations in PD and NDD area could even be used to solve the NLD task such as STS [28]. However since these methods are not specifically designed for NLD task, the performance is limited. As this thesis integrates text or word similarity methods at different NLP levels, some work in this area is reviewed.

2.1 Paraphrase Detection

PD is one of the relevant technique among existing methods to solve the NLD task. Four types of methods have been proposed for PD [13]: unsupervised string and semantic similarity-based methods [28], unsupervised lexical similarity-based methods [5], supervised context similarity based methods [50], and methods based on linguistic analysis of comparable corpora [26]. The unsupervised STS system proposed by Islam et al., and the supervised RAE system proposed by Socher et al. are utilized as the baseline methods of NLD task.

Socher et al. [50] proposed a method called RAE, which is the state-of-the-art supervised measure since 2011. They present a joint model that incorporates the similarities between both single word features as well as multi-word phrases extracted from the nodes of parse trees. The RAE is a recursive neural network. It learns feature representations for each node in the tree such that the word vectors underneath each node can be recursively reconstructed. These feature representations are used to compute a similarity matrix that compares both the single words as well as all non-terminal node features in both sentences. Then, a new dynamic pooling layer which outputs a fixed-size representation is applied. Finally, a classifier is utilized to classify whether a sentence pair are paraphrases or not.

Islam et al. [28] proposed a method that firstly measure the string similarity and word semantic similarity, and then use an optional common-word order similarity function to incorporate syntactic information. Finally, the text similarity is derived by combining string similarity, semantic similarity and common-word order similarity with normalization. The contents above the final similarity threshold are regarded as paraphrase. Their implementation STS is the state-of-the-art unsupervised PD system.

Unsupervised lexical similarity[5] utilized the simple word N-gram (n=1, 2, 3, 4) overlap measure in the context of paraphrase learning. A hierarchical complete-link clustering is produced based on the word N-gram similarity measure.

W.B Dolan, et al. [16] proposed a method to find and extract monolingual paraphrases from massive comparable news stories. They compare Edit Distance with an heuristic derived from Press writing rules. The experiments show that the data produced by the Edit Distance is clean and easy to align. However, using word error alignment rate results reveal that both techniques perform similarly.

More extensive methods that rely on context similarity measures such as sentence alignment method for monolingual comparable corpora [4] are also proposed. These methods find sentence alignments in comparable corpora by considering sentence contexts after semantically aligning equivalent paragraphs. But these methods rely on supervised learning techniques, which need large volume of training data that may be scarce and difficult to obtain.

Hatzivassiloglou et al. [26] made further analysis through exploring heavy linguistic features combined with machine learning methods to propose a new text similarity method. Since it is a supervised method which relies heavily on valuable linguistic resources, the methods are of limited practicality, especially when considering multiple languages.

2.2 Near-duplicate Detection

Another task that related to NLD is ND, especially the sub-domains NDD and NTD. However, to the best of our knowledge, there is no implementation available in these areas that fit NLD task.

In NDD area, two state of the art duplicate detection algorithms exist [59]: simhash [9] and shingle-based methods [8]. Broder et al. define sequences of tokens of length w that appear in a document as shingles, and measure the similarity of two documents by calculating the number of common shingles. The simhash algorithm maps a feature space to a fixed-size fingerprint. The process involves calculating a hash that represents each document and then detecting near duplicates by identifying documents that have similar hashes.

In NTD area, the research focuses on the reduction of text comparison time in large-scale text databases. Clustering and summarization techniques are applied. As for the text similarity measures, even the state-of-the-art NTD methods are utilizing very basic methods. Sun et al. [53] utilizes LCS, which returns high scores when two text strings share a large portion of identical substrings. Earlier work, such as the Simfinder proposed by Gong et al. [24], utilizes word N-gram to measure the similarity among texts.

2.3 Stream-based Similarity Measures

A stream-based similarity method provides a straightforward relatedness that reflecting how similar two strings appear. It operates on string sequences and character composition, and measures similarity or dissimilarity (distance) between two text strings for approximate string matching or comparison. Two groups of stream-based similarity measures are commonly utilized in text mining area: character-based similarity measures and term-based similarity measures.

2.3.1 Character-based Similarity Measures

Character-based similarity divide a string into single characters and measures relatedness by considering the order and frequency of each characer.

Longest Common SubString (LCS) algorithm is widely used as a character-based similarity measure. It considers the similarity between two strings based on the length of contiguous chain of characters that exist in both strings. N-gram is a sub-sequence of n items from a given sequence of text.

N-gram similarity algorithms compare the character units in two strings. Distance is computed by dividing the number of similar n-grams by maximal number of n-grams. V. Keselj, et al. [32] made further studies on n-gram similarity measures and propose the CNG distance to measure the distance between two strings.

D. Levenshtein, et al. [25] defines distance between two strings by counting the minimum number of operations needed to transform one string into the other, where an operation is defined as an insertion, deletion, or substitution of a single character, or a transposition of two adjacent characters.

Another character-based similarity measure is proposed by S. Waterman, et al [48]. They perform a local alignment to find the best alignment over the conserved domain of two sequences. It is useful for dissimilar sequences that are suspected to contain regions of similarity or similar sequence motifs within their larger sequence context.

2.3.2 Term-based Similarity Measures

Term-based similarity divide a string into individual terms and use each term as the smallest unit to make calculation and comparison.

Cosine similarity is a measure of similarity between two vectors (bag of terms) of an inner product space that measures the cosine of the angle between them. As the idea is simple and straightforward and easy to implement, Cosine similarity is commonly used as a baseline method in similarity measurements.

Jaccard similarity [30] is also a term-based similarity measure, which is computed as the number of shared terms over the number of all unique terms in both strings.

Dice's coefficient [15] is defined as twice the number of common terms in the compared strings divided by the total number of terms in both strings. There are some other coefficient methods that are actually variations of Dice's coefficient measure, such as Overlap coefficient and Matching Coefficient, which are also term-based similarity measures.

2.4 Corpus-Based Similarity Measures

Corpus-Based similarity is a semantic similarity measure that determines the similarity between words according to information gained from large corpora [23].

Latent Semantic Analysis (LSA) is the most popular technique of Corpus-Based similarity. LSA assumes that words that are close in meaning will occur in similar pieces of text. A matrix containing word counts per paragraph is constructed from a large piece of text.

Explicit Semantic Analysis (ESA) is a measure used to compute the semantic relatedness between two arbitrary texts. The Wikipedia-Based technique represents terms (or texts) as high-dimensional vectors; each vector entry presents the TFIDF weight between the term and one Wikipedia article. The semantic relatedness between two terms (or texts) is expressed by the cosine measure between the corresponding vectors.

Normalized Google Distance (NGD) and Google Tri-gram Method (GTM) are similarity measures based on the corpus formed by collecting the search behaviors on Google search engine. Considering that keywords with the same or similar meanings in a natural language sense tend to be "close" in units of Google distance and words with dissimilar meanings tend to be farther apart, these corpus-based similarity also measure semantic relatedness.

2.5 Knowledge-Based Similarity Measures

Knowledge-Based Similarity is one of semantic similarity measures that bases on identifying the degree of similarity between words using information derived from semantic networks [39].

2.5.1 WordNet Similarity Measures

WordNet [40] is the most popular semantic network in the area of measuring the knowledge-based similarity between words; WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. Complex word relationships

are organized as a hierarchical structure in WordNet. It could reflect relationships between concepts such as is-a-kind-of, is-a-specific example-of, is-a-part-of, is-the-opposite-of. Benifit from WordNet, quite a lot studies on semantic similarity area has been carried out, and several semantic similarity measures are proposed.

Resnik, et al. [45] proposed to measure the relatedness using the information content (IC) of the Least Common Subsumer (most informative subsumer). The relatedness value will always be greater-than or equal-to zero. The upper bound on the value is generally quite large and varies depending upon the size of the corpus used to determine information content values.

Lin, et al. [37] augment the information content of the Least Common Subsumer with the sum of the information content of concepts A and B themselves. The measure scales the information content of the Least Common Subsumer by the sum.

Jiang, et al. [31] also utilized the information content of the Least Common Subsumer with the sum of the information content of concepts, and then they take the difference of this sum and the information content of the Least Common Subsumer to measure the similarity.

There are also measures that utilize the path length from one word to another word of WordNet to calculate concepts relatedness. Leacock, et al. [35] propose a measure that returns a score denoting how similar two word senses are, based on the shortest path that connects the senses and the maximum depth of the taxonomy in which the senses occur. Wu, et al. [60] calculate a score denoting how similar two word senses are, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer.

2.5.2 Wikipedia Similarity Measures

Wikipedia provides a knowledge base for computing word relatedness in a more structured fashion than a search engine and with more coverage than WordNet [21]. The strength of Wikipedia lies in its size, which could be used to overcome current knowledge bases' limited coverage and scalability issues.

Mohamed, et al [41] propose a system that utilize Wikipedia features (articles,

categories, Wikipedia category graph and redirection) to compute semantic relatedness between words. Their approach is preceded by a pre-processing step to provide for each category pertaining to the Wikipedia category graph a semantic description vector. Next, for each candidate word, they collect its categories set using an algorithm for categories extraction from the Wikipedia category graph and then using vector similarity metrics to measure semantic relatedness.

M. Strube, et al. created Wikirelate system [51], which is based on category structure. Their system works as follows: given the word pairs (w_i, w_j); they first retrieve the Wikipedia pages which they refer to. Then, they run through the category tree to extract the categories that the pages belong to. Finally, they compute relatedness based on the pages extracted and the paths connecting categories in the category taxonomy.

Gabrilovich, et al. [21] use Wikipedia articles as index documents since Wikipedia covers a wide range of topics, while each article is focused on one topic. Each Wikipedia concept is represented as a vector of words that occur in the corresponding article. Entries of these vectors are assigned weights using TFIDF scheme.

WikiWalks proposed by Yeh, et al. [61], explores several methods for using the link structure of Wikipedia with random walks based on Personalized PageRank, which is computed for each text fragment. They map each input word to its respective nodes in the graph to create its teleport vector. Personalized PageRank is then executed to compute the stationary distribution for each word, using their respective teleport vectors. Finally, the stationary distributions for each word pair are scored with cosine similarity measure.

2.6 Regression and Classification Model

This thesis performs several independent similarity methods that measure text and word similarity from different perspectives at different NLP levels. All the analysis results are regarded as independent features, and are integrated by applying a classification method to classify the candidate sentence pairs into true non-uniform language instances and false instances. The state of the art regression and classification methods are reviewed in this section.

2.6.1 SVM

The foundation of Support Vector Machines (SVM) has been developed by Vapnik [57], and has gained popularity due to its many attractive, analytic and computational features, and promising empirical performance.

SVM is developed to solve the classification problem, in which it is shown that the generalization error is bounded by the sum of the training set error and a term depending on the Vapnik-Chervonenkis (VC) dimension of the model. Later on, Vapnik, et al. extended the SVM to the domain of regression problems [56]. Then SVM is widely applied in regression domain and quite a lot measures based on SVM have been proposed, such as probabilistic framework for SVM classification [22], improved SVM regression using mixtures of kernels [49], and Least Squares Support Vector Machines (LSSVM) [54], etc.

2.6.2 Naive Bayes Model

Naive Bayes is a probabilistic based theory that has been studied extensively since the 1950s, and remains a popular (baseline) method for text classification. It is highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. The idea and implementation of Naive Bayes model is simple. Yet, with appropriate preprocessing, Naive Bayes measure is competitive in this domain with more advanced methods such as SVM.

Several researchers have tried to improve Naive Bayes by deleting redundant attributes [34], or by extending it to incorporate simple high-order dependencies [20]. P. Domingos, et al. [17] review these approaches in some detail, and made improvement by "relaxing the independence assumption have had mixed results."

2.6.3 KNN

Unlike most other regression and classification approaches, the K-Nearest Neighbor (KNN) measure does not build a model from the training data. Instead, it turns training examples into vectors in a high dimensional feature space. KNN is widely applied in document classification area.

Given a test document d, KNN find the k nearest neighbors of d among all the training documents, and score the category candidates based on the category of k neighbors. The similarity of d and each neighbor document is the score of the category of the neighbor document. If several of the k nearest neighbor documents belong to the same category, then the sum of the score of that category is the similarity score of the category in regard to the test document d. By sorting the scores of the candidate categories, system assigns the candidate category with the highest score to the test document d.

Chapter 3

Text Similarity Methods

This chapter describes text similarity methods we experimented with in this work. These algorithms are chosen of different characteristics for better understanding of their benefits and limitations in this context. We combined these text similarity methods with different thresholds to extract similar sentence pairs from a document. These sentence pairs are regarded as non-uniform language candidates and the process of generating such sentence pair is the first stage of our non-uniform language detection solution.

3.1 Cosine Similarity

Cosine similarity is one of the most popular text similarity algorithms. It measures the degree of similarity of two documents as the correlation between their corresponding vector representations, which can be quantified as the cosine of their angle. Given two documents $d1$ and $d2$, their cosine similarity is:

$$\cos(\vec{d_1}, \vec{d_2}) = \frac{\vec{d_1} \bullet \vec{d_2}}{\| \vec{d_1} \| \| \vec{d_2} \|} \tag{3.1}$$

Despite the fact that it ignores the relative order of the words in the document, it offers a competitive baseline for text similarity.

3.2 Longest Common Subsequence

Longest Common Subsequence (LCS) is another widely employed technique to measure similarity between texts. It measures the total length of the longest matching substrings in both texts, where these substrings are allowed to be non-contiguous as long as they appear in the same order [10, 27]. While the original algorithm was applied to find substrings of characters, a natural extension is to consider it for words, i.e. the longest common substring has to be composed by a sequence of full

words only. The final similarity score can be obtained by dividing the number of words of the longest common subsequence by the length in words of the shortest document under comparison.

3.3 Google Tri-gram Method

GTM is an unsupervised corpus-based approach for measuring semantic relatedness between texts [29]. GTM uses unigrams and trigrams from the Google Web 1T N-gram corpus [7] to find the relatedness of a pair of words, and extends the word relatedness method to measure the document relatedness.

$$\mathrm{GTM}(\omega_1,\omega_2) = \begin{cases} \dfrac{\log \frac{\mu_T(\omega_1,\omega_2)C_{\max}^2}{C(\omega_1)C(\omega_2)\min(C(\omega_1)C(\omega_2))}}{-2\times\log\frac{\min(C(\omega_1),C(\omega_2))}{C_{\max}}} & \text{if } \log\frac{\mu_T(\omega_1,\omega_2)C_{\max}^2}{C(\omega_1)C(\omega_2)\min(C(\omega_1),C(\omega_2))} > 1 \\ \dfrac{\log 1.01}{-2\times\log\frac{\min(C(\omega_1),C(\omega_2))}{C_{\max}}} & \text{if } \log\frac{\mu_T(\omega_1,\omega_2)C_{\max}^2}{C(\omega_1)C(\omega_2)\min(C(\omega_1),C(\omega_2))} \leq 1 \\ 0 & \text{if } \mu_T(\omega_1,\omega_2) = 0 \end{cases}$$

The Google Web 1T N-gram corpus, contributed by Google Inc., contains English word n-grams (from unigrams to 5-grams) and their observed frequency counts calculated over one trillion words from web page texts collected by Google in January 2006 [7].

Table 3.1: Notation used for GTM word relatedness measurement

Notation	Description
$C(\omega)$	Count of the word ω.
$\mu_T(\omega_1,\omega_2)$	Mean frequency of trigrams which either start with ω_1 and end with ω_2, or start with ω_2 and end with ω_1.
$\sigma(a_1,\ldots,a_n)$	Standard deviation of numbers $a_1,\ldots,a_n$
$C_{\max}$	Maximum frequency among all unigrams.

GTM similarity can be computed online.

Chapter 4

Non-uniform Language Detection

A framework that consists of three stages is proposed as a solution of the NLD task. The first stage extracts candidate sentence pairs that have high text similarity within a document. The second stage performs comprehensive analysis on each candidate sentence pair. The analysis is performed at lexical, syntactical, semantic, and pragmatic levels and multiple NLP resources such as POS tagging, WordNet, GTM, and Flickr are utilized. The final stage integrates all the analysis results by applying a classification method based on SVM to classify the candidate sentence pairs as True or False non-uniform language cases.

The overview of the proposed solution is shown in Figure 4.1.

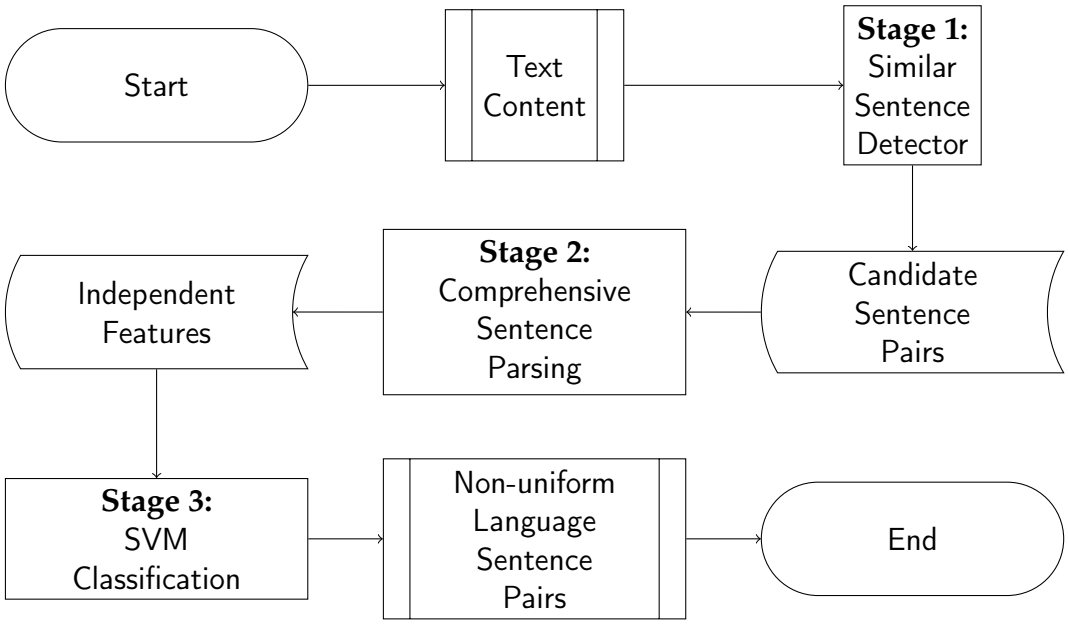

Figure 4.1: Framework of our NLD solution

4.1 Stage 1: Similar Sentences Detection

To extract the candidate sentence pairs, three text similarity algorithms are integrated at the sentence level. GTM, LCS, and Cosine Similarity are used under certain thresholds to filter the pairs based on the semantic meaning, sentence structure, and syntax, respectively. Each similarity method scales from 0 to 1. A higher value means the stronger relationship between the compared sentences. By combining these methods, we could extract similar sentences either appears similar, or related in semantic or syntax level.

The filtering thresholds are set by running experiments at the sentence level on the iPhone user manual [3]. The algorithm to set the thresholds of different similarity measures is shown in Algorithm 1.

We utilized a sentence detector [44] to divide the text of the manual into a sentence set (Line 2) and applied the similarity thresholds setting algorithm on this sentence set. We separately ran the algorithm three times to set the threshold sets for GTM, LCS, and Cosine. The thresholds were set for each average length of sentence pair. The average size starts from two and is increased by one each time once the certain threshold for the current size is set. We discovered that once the sentence length reaches ten or higher value, the thresholds vary little. So we made the algorithm stop when the threshold for size of ten is set (Line 6).

For each sentence pair size, the algorithm starts by asking the user to input an initial similarity threshold and an increasing step (Line 4-5). An initial threshold range is generated based on the user setting. The lower bound of the range is T and the upper bound of the range is T+*Step* (Line 9-10). Then the algorithm would loop over all the sentence pairs (Line 11-20) and filter some of them based on the current similarity threshold setting and output the pairs within the current threshold range (Line 14-16). Once the loop is finished, the algorithm increases the threshold based on the step value set by the user (Line 21). Then the algorithm would loop over all the sentence pairs again using the new threshold to narrow down the output. The similarity of sentence pairs above the previous threshold and below the current threshold are captured and analyzed (Line 22). If they consist of all false non-uniform language candidates, we repeat the experiment with a higher threshold to filter more false candidates. Once we discover that a true candidate is filtered by

Input : Initial threshold T,
Initial threshold increasing step $Step$
Data: User Manual
Output: ThresholdSizeList $\{(T_1, Size_1), ...\}$

begin
  $S[n] \longleftarrow SentenceDetector$(User Manual) /*Split the manual into n sentences*/
  $Size \longleftarrow 2$ /*$Size$: Average length of a sentence pair*/
  $T \longleftarrow$Initial Threshold from user
  $Step \longleftarrow$Initial increasing step from user
  while $(Size \leq 10)$ **do**
    $C \longleftarrow \emptyset$ /*Initial the output sentence container.*/
    do
      $T_{low} \longleftarrow T$
      $T_{up} \longleftarrow T + Step$
      for *(i=0; i<n; i++)* **do**
        for *(j=0; j<n; j++)* **do**
          $AveSize \longleftarrow (S[i] + S[j])/2$
          if $AveSize \in [Size - 1, Size)$ **then**
            if $(T_{low} \leq Sim(S[i], S[j]))$ **and** $(Sim(S[i], S[j]) \leq T_{up})$ **then**
              $C \xleftarrow{add} (S[i], S[j])$
            end
          end
        end
      end
      $T \longleftarrow T+Step$
    while $(Check(C) = True)$
    */*Done by human. Return True when all the sentence pairs are not non-uniform language.*/;*
    $ThresholdSizeList \xleftarrow{add} (T_{low}, Size)$
    $Size$++;
  end
end

Algorithm 1: Similarity thresholds setting algorithm

the current threshold, we stop increasing and set the prior value as the threshold to maximize the recall ratio. The whole experiment is repeated for different sentence pair lengths. The final thresholds for different similarity methods are shown in Figure 4.2.

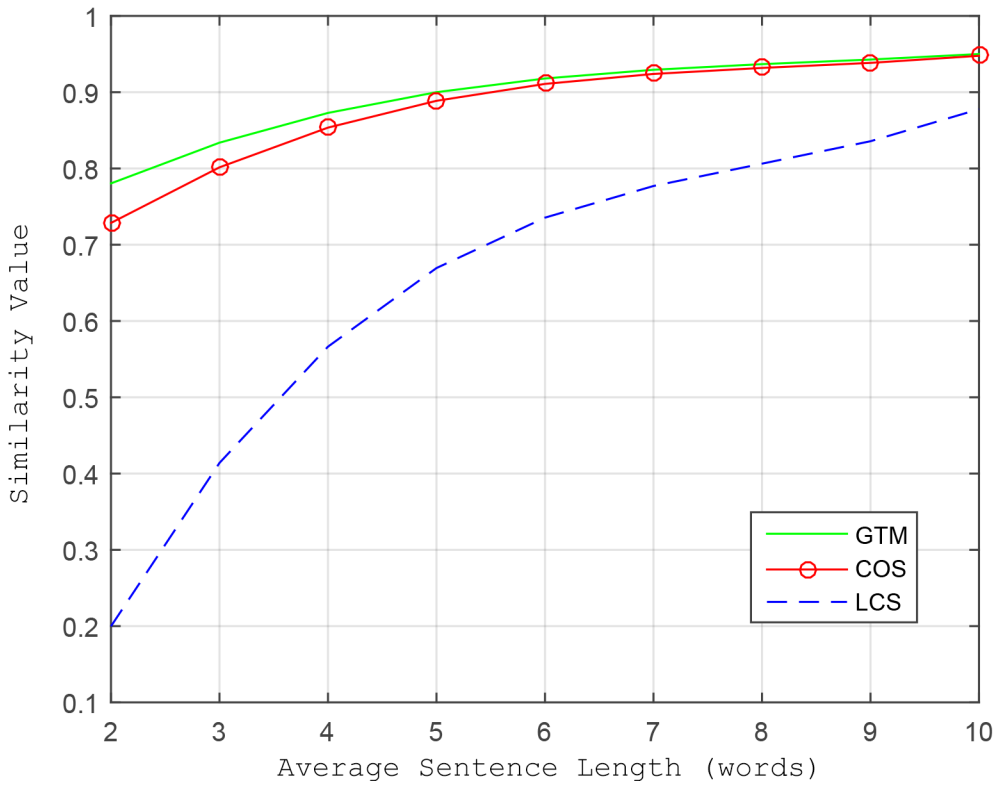

Figure 4.2: Candidate Sentence Detection Threshold

To filter the sentence pairs, we applied the thresholds of the three text similarity algorithms. For example, assume there are two sentences of nine-word length on average. The similarity scores of this pair have to be above the GTM, LCS and Cosine thresholds (which are 0.943, 0.836, and 0.932, according to Figure 4.2) to make it a candidate sentence pair.

By applying the thresholds shown in Figure 4.2, candidate pairs could be detected in reasonable scale and good recall ratio. As for precision, around 30% of candidates generated in this stage are true non-uniform language, and the remaining of them are false candidates that are supposed to be filtered in the second stage.

4.2 Stage 2: Sentence Parsing

In this stage, analysis is made on each candidate pair generated in the previous stage. The analysis aims to figure out whether the two sentences describe the same object/operation using different words/writing style (True non-uniform language) or that they just appear similar but actually describe different things.

4.2.1 Part-of-Speech Tagging Analysis

To begin with, POS tags are added for each candidate pair using NLTK [42] tool. By applying POS analysis, we are able to gain a grammatical view over the sentences as shown in Table 4.1.

Candidate Sentence Pair with POS Tag	Ground Truth
Link your device to iTunes stores /NNP /PRP /NN /TO /NNS /NNS	True
Link your device to iTunes store /NNP /PRP /NN /TO /NNS /NN	Candidate
go to settings > general > accessibility > audio /VB /TO /NNS /SYS /JJ /SYS /NN /SYS /NN	False
go to settings > general > accessibility > vedio /VB /TO /NNS /SYS /JJ /SYS /NN /SYS /NN	Candidate
Hold the power button for two seconds to shutdown the device /VB /DT /NN /NN /IN /NN /NNS /TO /NN /DT /NN	True
Hold the power button for two seconds to shut down the device /VB /DT /NN /NN /IN /NN /NNS /TO /VBN /RB /DT /NN	Candidate
Hold the power button for two seconds to turn on the device /VB /DT /NN /NN /IN /NN /NNS /TO /VBN /IN /DT /NN	True
Hold the power button for two seconds to shut down the device /VB /DT /NN /NN /IN /NN /NNS /TO /VBN /RB /DT /NN	Candidate

Table 4.1: POS Analysis of Candidate Sentence Pairs

As Table 1 shows, some differences in sentence content could be reflected by POS tags, and some could not. Thus, it is necessary to make further syntactic and semantic analysis to distinguish true candidates from false ones.

After POS tagging, the different parts within each candidate pair are captured. Based on the extracted different parts with POS tags, we categorized the different

POS tags into the following groups shown in Table 4.2. The different POS tags are mapped to each category label and saved as an independent feature.

Label	Description	Example
1	Equal length, same POS tag	/NN vs./NN /VB vs./VB
2	Equal length, plural noun with singular noun	/NN vs./NNS
3	Equal length, different POS	/NN vs./VB
4	Unequal length, extra article	/NN vs./DT/NN
5	Unequal length, extra conjunction	/NN vs./CC/NN
6	Unequal length, extra adjective	/NN vs./JJ/NN
7	Other POS tag types.	/NN vs. N/A

Table 4.2: POS Tag Categorizing

4.2.2 Character N-gram Analysis

In the N-gram analysis, the distance between the different parts of each candidate pair is calculated in terms of unigram, bigram and trigram similarity. The character N-gram frequencies with a window sized from 1 to 3 is firstly calculated. Then, the N-gram distance based on the frequencies is calculated using equation (4.1):

$$d(f_1, f_2) = \sum_{n \in dom(f_1) \cup dom(f_2)} \left(\frac{f_1(n) - f_2(n)}{\frac{f_1(n)+f_2(n)}{2}}\right)^2 \tag{4.1}$$

$dom(f_i)$ is the domain of function f_i.

In the equation above, n represents a certain N-gram unit. $f_i(n)$ represents the frequency of n in sentence i *(i=1,2)*. If n does not appear in sentence i, $f_i(n) = 0$. So the lower bound of the N-gram distance would be 0, which means the two units to be compared are exactly the same. The higher the value of N-gram distance, the larger the difference, and it has no upper bound.

4.2.3 WordNet Lexical Relation Analysis

When comparing a pair of candidate sentences, if the only different parts (word or phrase) are synonymous to each other, chances are great that the two sentences try to convey the same meaning but use different expression, which means this pair is our target non-uniform language. On the other hand, if the different parts of a

candidate pair are not related at the lexical level, then it is reasonable to assume that this pair is describing different things; thus they might not be non-uniform language.

Considering this feature, WordNet is utilized to analyze the lexical relationship within each candidate pair to indicate whether they are synonymous to each other. Table 4.3 provides some examples using WordNet analysis.

Different Content Pair	**Is Synonym**
(stores, store)	True
(audio, gray scale)	False
(shutdown, shut down)	True
(turn on, shut down)	False

Table 4.3: Example of Synonym Detection using WordNet

As we may concern, WordNet organizes words in a hierarchical structure, and indicates lexical relationships such as hypernymy, hyponymy, antonymy, and synonym. Lexical relatedness can be measured by calculating the path from one node to another node. There are three reasons we only applied synset and only considered whether words belonged to the same synset or not. Firstly, we discover that whether a similar sentence pair is an instance of nonuniform language depends on whether the different words are synonyms, rather than having other relationships such as hypernymy, hyponymy, and antonymy. Therefore, we do not deem necessary to include these relationships into our analysis. Secondly, the path relatedness measure in WordNet covers a large range of relationship, such as "is-a-kind-of", "is-a-specific example-of", "is-a-part-of", and "is-the-opposite-of". The relatedness is reflected by a normalized value scale from 0 to 1. Such continuous value may not able to provide explicit numerical evidence compared to boolean value, and it is not easy to find an accurate threshold for classification. Thirdly, even though we find a certain threshold, we are unable to distinguish the relationships behind the relatedness value. For example, given a similar sentence pair:

```
if the photo hasn't been downloaded yet, tap the download notice first.
if the video hasn't been downloaded yet, tap the download notice first.
```

The sentence pair above is not a non-uniform language instance. However, the relatedness score between `photo` and `video` given by WordNet is 0.6 [2], which is fairly high compared to other word pairs. Yet we do not know these words are related in the way 'photo is a kind of video', or 'photo is a part of video', or 'photo and video are examples of media content'. Therefore, we might make wrong judgments based on this similarity score. However, using synset information, we could know that these words are not synonyms and thus probably not suggesting a non-uniform language instances. In order to remove the redundant and noisy information, and provide explicit numerical evidence for the machine learning stage, we apply synset of WordNet only and utilize boolean value to represent the relationship.

4.2.4 Flickr Related Concept Analysis

In some cases, word to word relatedness exists and goes beyond dictionary definitions. This kind of relatedness is categorized as metonymy, in which a thing or concept is called not by its own name but rather by the name of something associated in meaning with that thing or concept [33]. It is also necessary to detect metonymy in technical writings. To detect metonymy is actually a task in the pragmatic level of NLP area.

Flickr [19] is a popular photo sharing website that supports time and location metadata and user tagging for each photo. Since the tags are added by humans and aim to describe or comment on a certain photo, the tags are somehow related from a human perspective. As a result, Flickr becomes a large online resource which is able to indicate metonymies among text.

Flickr made available statistical information about their dataset that can be used to query related concepts of a certain word or phrase online. We utilized this resource to detect whether the different parts within a candidate sentence pair are related at the pragmatic level. A boolean value that indicates metonymy relationship is returned and saved. This result is regarded as a feature data of the NLD analysis. Table 4.4 gives some examples of relatedness that could be discovered in this stage.

Different Content	Is Metonymy
aeroplane, A380	True
film, hollywood	True
apple, iPhone	True
audio, grayscale	False

Table 4.4: Example of Metonymy Detection using Flickr

4.3 Stage 3: SVM Classification

All the metrics described above are regarded as features of our candidate sentence pairs. To make a comprehensive judgment based on these analysis from different perspectives, a classification method based on SVM [57] is applied. We implemented the SVM classification using "e1071" package [58] in R.

By inputting a group of labeled data as the training set and then classifying unlabeled data (candidate pairs extracted in stage 1) into true or false classes, a final judgment is given. If an instance is predicted as a true candidate, then the system outputs it as a nonuniform language instance; otherwise, the system would remove it from the candidate list. In the next chapter, several experiments are described with data and results using the methodology introduced in this section.

Chapter 5

Experiments and Evaluation

In this chapter, experiments under the proposed solution were carried out. Experiment data and results are described. Some evaluation methods and baseline results are provided.

5.1 Experiment Data

We downloaded smart phone user manuals of iPhone [3], LG [36] and Samsung [46] online and extracted 325 candidate sentence pairs (650 sentences) in stage 1 as our data set (DS). Before applying the sentence parsing and classification experiment, each candidate sentence pair in DS was labeled by three different experts as true or false. Then the ground truth for each instance is generated by experts' voting. For each candidate instance, if two or more experts label it as true, then the ground truth for this instance is true; otherwise, the ground truth label for this instance would be false. Some statistics from the manuals are shown in Table 5.1.

Data Source	Data Volume (Pages)	Candidate Pairs (True, False)
iPhone	196	208 (102,106)
LG	274	54 (16,38)
Samsung	190	63 (32,31)

Table 5.1: Experiment Data Volume and Distribution

Considering the data distribution in the original document for the NLD task is unbalanced in terms of true/false instances, the training set (DS_{train}) and test set (DS_{test}) for the classification model was formed also unbalanced. DS_{train} was formed by randomly selecting 200 instances from DS and the remaining 125 instances were made as DS_{test}, so the portion of true/false instances would be different in DS_{train} and DS_{test}. A classification method based on SVM was applied to classify the candidates into true class or false class.

5.2 Main Measurement Criteria

In this set of experiments, we used *Precision*, *Recall*, *Accuracy*, and $F1$ to measure the performance of the solution against the ground truth. To measure precision of the system, we used the following formula:

$$P_s = \frac{\mid r_s \cap g_s \mid}{r_s} \tag{5.1}$$

where r_s is the set of returned non-uniform language instances detected by the system, and g_s is the set of real non-uniform language instances that must be returned according to gold standard.

We calculate the recall ratio of NLD with the same parameters above by the following formula:

$$R_s = \frac{\mid r_s \cap g_s \mid}{g_s} \tag{5.2}$$

Accuracy represents how well a binary classification test correctly identifies a condition. It calculate the proportion of true results (both true positives and true negatives) among the total number of cases examined by the following formula:

$$Accuracy_s = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \tag{5.3}$$

where T_p is set of true non-uniform language that detected correctly by the system, T_n is the set of false non-uniform language that filtered correctly by the system, F_p is the set of false non-uniform language that incorrectly detected by the system, and F_n is the set of real non-uniform language that failed to be detected by the system.

$F1$ measure is the harmonic mean of precision and recall, which represents an overall performance as well as the retrieval power of a system:

$$F1_s = \frac{2 \cdot R_s \cdot P_s}{R_s + P_s} \tag{5.4}$$

5.3 Evaluation on Human Experts

To address the ground truth for each test case, we invited three human expert raters to manually label each case as true or false non-uniform language instance. The gold standard is then generated based on three human experts' voting. A test case is assigned as true non-uniform language when two or more raters labeled it as true. Otherwise, the test case is assigned as false non-uniform language. The performances of human experts was evaluated against the gold standard in terms of *Precision, Recall, Accuracy,* and *F1-measure*, using the whole data set DS. We also applied a Kappa test to calculate the different judgments that different human experts could make against a same test case, and utilized this value to evaluate how difficult the NLD task is.

5.3.1 Gold Standard

A confusion matrix evaluating the experts against the gold standard is generated as follow:

Experts	TP	TN	FP	FN
Expert 1	130	161	20	14
Expert 2	99	164	51	11
Expert 3	125	166	25	9

Table 5.2: Confusion Matrix of Experts Judgments

In the table above, TP is the set of true non-uniform language that the expert recognized correctly, TN is the set of false non-uniform language that filtered correctly by the expert, FP is the set of false non-uniform language that incorrectly recognized by the expert, and FN is the set of real non-uniform language that failed to be recognized by the expert. Based on these value, the performance of each expert is evaluated in terms of *Precision*, *Recall*, *Accuracy*, and *F1-measure* as follow:

Experts	Precision	Recall	Accuracy	F1
Expert 1	**86.67**	90.27	89.54	**88.43**
Expert 2	66.00	90.00	80.92	76.15
Expert 3	83.33	**93.28**	**89.54**	88.03

Table 5.3: Evaluation of Experts Judgments

5.3.2 Fleiss' Kappa Test

To reveal the difference of three human judgments, the Fleiss' Kappa Value [18] is calculated. Fleiss' Kappa is an extension version of Cohen's Kappa [11]. Unlike Cohen's Kappa, which only measures the agreement between two raters, Fleiss' Kappa measures the agreement among three or more raters.

Agreement can be thought of as follows, if a fixed number of people assign numerical ratings to a number of items then the kappa will give a measure for how consistent the ratings are. The kappa can be defined as follow:

$$\kappa = \frac{\overline{P} - \overline{P_e}}{1 - \overline{P_e}} \tag{5.5}$$

The factor $1 - \overline{P_e}$ gives the degree of agreement that is attainable above chance, and $\overline{P} - \overline{P_e}$ gives the degree of agreement actually achieved above chance. The value of $\overline{P}$ and $\overline{P_e}$ could be calculated as follow:

Let N be the total number of subjects, let n be the number of ratings per subject, and let k be the number of categories into which assignments are made. The subjects are indexed by $i = 1, ...N$, and the categories are indexed by $j = 1, ...k$. Let n_{ij} represent the number of raters who assigned the i-th subject to the j-th category.

First calculate p_j, the proportion of all assignments which were to the j-th category:

$$p_j = \frac{1}{N \cdot n} \sum_{i=1}^{N} n_{ij} \tag{5.6}$$

Then calculate P_i, which is the extent to which raters agree for the i-th subject. In other words, P_i reflects how many $rater - rater$ pairs are in agreement, relative to the number of all possible $rater - rater$ pairs. The P_i is calculated by the following formula:

$$P_i = \frac{1}{n \cdot (n-1)} \sum_{j=1}^{k} n_{ij} \cdot (n_{ij} - 1) \tag{5.7}$$

Next, we calculate $\overline{P}$, the mean of P_i, by the following formula:

$$\overline{P} = \frac{1}{N} \sum_{i=1}^{N} P_i \tag{5.8}$$

Finally, we calculate $\overline{P_e}$ by the following formula:

$$\overline{P_e} = \sum_{j=1}^{k} P_i^2 \tag{5.9}$$

By substitute the value of $\overline{P}$ and $\overline{P_e}$ into equation 5.5, we could get the final kappa value κ.

In our case, we have 3 experts (the raters number n is 3), each expert labeled 325 candidate pairs (the subject volume N is 325), each candidate pair is labeled either 0 or 1 (the value of category k is 2). By applying formulas above in this subsection, we get the final Fleiss' Kappa Value: 0.545.

The Kappa Interpretation Model [18] gives the following table for interpreting κ values:

κ	**Interpretation**
< 0	Poor agreement
0.01 - 0.20	Slight agreement
0.21 - 0.41	Fair agreement
0.41 - 0.60	Moderate agreement
0.61 - 0.80	Substantial agreement
0.81 - 1.00	Almost perfect agreement

Table 5.4: Kappa value interpretation table

According to the table above, the human experts could achieve a moderate agreement level (0.41 - 0.60) on the same data set in NLD task. In other words, the performances of three experts reveal that the NLD task is not simple, since there are many cases that are ambiguous and hard to make accurate judgments even for humans.

5.3.3 Upper bound Performance

Considering the NLD task is difficult as the Kappa test reveals that even human experts could not achieve high agreement on labeling test cases, we defined upper bound performance to evaluate our proposed solution as a complementary standard of our measure criteria.

As we can see in table 5.3, the best performance in terms of *Precision, Recall, Accuracy*, and *F1-measure* among the three experts are highlighted using bold font.

Even though the three experts are not able to represent the best performance of human beings, these values are indeed the result of human knowledge gained from three rather than one person. We claim that a NLD system is good enough if its performance is close to human performance. Thus, we define the upper-bound performance using the best combined performance of the three experts, and named it as UB method that shows in table 5.7.

5.4 Baseline Method

As we reviewed in chapter 2, NLD is a novel task but related to PD and NDD area. We found the most up to date solutions, including supervised and unsupervised implementations in PD area. And to our best knowledge, there is no implementation that fit the NLD task, available in NDD area. We tested the unsupervised measure STS [28], and the supervised measure RAE [50], using the data set DS. The results are regarded as the baseline performance of NLD task.

5.4.1 STS

STS is an unsupervised PD system, which utilized a corpus-based semantic similarity measure. Since STS is unsupervised, we ran STS using our data set DS without separate it into training set DS_{train} and testing set DS_{test}. The experiment shows that all the candidate sentences are recognized as non-uniform language instance by STS. The result is provided in table 5.5.

TP	TN	FP	FN	Precision	Recall	Accuracy	F1
150	0	175	0	46.15	100.00	46.15	63.16

Table 5.5: Evaluation of STS

5.4.2 RAE

Socher et al. proposed RAE in 2011. RAE is the state of the art measure in PD area, which can be used as an off-the-shelf supervised paraphrase detection tool. RAE can be trained using arbitrary texts, and to be used as a baseline method of NLD task.

To campare with our $NLDS$, We utilized the exact same data split to run RAE. That is, we utlize DS_{train} to train RAE, and then test on DS_{test}. The result is provided in table 5.6.

TP	TN	FP	FN	Precision	Recall	Accuracy	F1
58	0	67	0	46.40	100.00	46.40	63.39

Table 5.6: Evaluation of RAE

As we can see in table 5.6, all the candidates are recognized as true paraphrase by RAE, so we are not able to distinguish true and false non-uniform language instances within the candidates by running RAE.

5.5 Evaluation on NLDS

We named the system we proposed as Non-uniform Language Detecting System (NLDS). After defining the upper-bound (UB) and the baseline performances, we evaluated NLDS by training a SVM classifier on DS_{train}, and then performed classification and classification using the SVM classifier on DS_{test}. The result is shown in Table 5.7 as the NLDS method. The first row presentes the upper bound performance and the following two rows present the baseline performances.

To assess the importance of each feature utilized in the proposed framework, we performed a feature ablation study [12] on N-gram, POS analysis, GTM, and Flickr, separately on the DS_{test} data set. The results are listed together with the proposed NLDS method in Table 5.7.

Method	**R**(%)*	**P** (%)*	**A**(%)*	**F1**(%)*
UB	92.38	86.67	**89.54**	**88.43**
STS	**100**	46.15	46.15	63.16
RAE	**100**	46.40	46.40	63.39
Uni-gram	11.11	35.29	52.80	16.90
Bi-gram	44.44	61.54	64.00	51.61
Tri-gram	50.00	62.79	65.60	55.67
POS	77.78	72.77	78.40	76.52
GTM	85.18	59.74	68.80	70.23
Flickr	48.96	94.00	74.00	64.38
NLDS	80.95	**96.22**	88.80	87.93

Table 5.7: Evaluation of Classification Result

5.6 Student's T Test

A student's t-test [52] is any statistical hypothesis test in which the test statistic follows a Student's t-distribution if the null hypothesis is supported. It can be used to determine if two sets of data are significantly different from each other, and is most commonly applied when the test statistic would follow a normal distribution if the value of a scaling term in the test statistic were known. When the scaling term is unknown and is replaced by an estimate based on the data, the test statistic (under certain conditions) follows a Student's t distribution.

5.6.1 One-sample T Test

A One-sample T Test addresses whether the mean of a population has a value specified in a null hypothesis. In testing the null hypothesis that the population mean is equal to a specified value μ_0, the following formula is applied:

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}} \tag{5.10}$$

where $\bar{x}$ is the sample mean, s is the sample standard deviation of the sample, and n is the sample size. The degrees of freedom used in this test are $n - 1$.

In our case, we define the upper bound performance in terms of precision, recall ratio, accuracy, and F1-Measure, which are already known, as the value of μ_0. Then, a 5-fold cross validation using NLDS on DS data set is performed. 65 out of 325 instances are utilized as testing set, so the degree of freedom in this case is 64. The results are shown in Table 5.8:

Tested Parameter	μ_0**(%)**	$\bar{x}$**(%)**	s**(%)**	n	t
Recall	92.38	86.55	4.60	65	11.47
Precision	86.67	92.24	3.92	65	-10.23
Accuracy	89.54	88.61	3.96	65	-1.88
F1	88.43	89.24	3.61	65	1.82

Table 5.8: Student's T-test results on UB and NLDS

According to the T-Distribution Table [1], we find that with a confidence factor $\alpha = 0.05$, $d = 64$, the threshold T is 2. From the statistic perspective, with a 95% confidence degree, we claim that the precision of NLDS is significantly better than UB, and the recall ratio of NLDS is significantly lower than UB. However, there

is no significant difference between NLDS and UB in terms of Accuracy and F1-Measure. Therefore, the overall performance of NLDS is able to achieve the upper bound performance that we defined for NLD task.

5.6.2 Two-sample Unpaired T Test

The independent (unpaird) two-samples T-test is used when two separate sets of independent and identically distributed samples are obtained, one from each of the two populations being compared. Such two-sample T-test could reveal whether the two sets differ from each other.

The independent two-sample t-test can be measured using the following formula:

$$t = \frac{\bar{X_1} - \bar{X_2}}{S_{X_1X_2} \cdot \sqrt{\frac{1}{n}}} \tag{5.11}$$

where

$$S_{X_1X_2} = \sqrt{S_{X_1}^2 + S_{X_2}^2} \tag{5.12}$$

Here, $S_{X_1X_2}$ is the grand standard deviation of group 1 and 2, $S_{X_1}^2$ and $S_{X_2}^2$ are the unbiased estimators of the variances of the two samples. For significance testing, the degrees of freedom for this test is $2n - 2$ where n is the number of participants in each group.

We applied the two-sample unpaired t-test to compare the performance between NLDS and the baseline method STS. A 5-fold cross validation is performed. 65 out of 325 instances are utilized as testing set, so the degree of freedom in this case is 128. The results are shown in table 5.9:

Tested Parameter	$\bar{X_{NLDS}}$**(%)**	$\bar{X_{STS}}$**(%)**	$S_{X_{NLDS}X_{STS}}$**(%)**	$t_{(NLDS,STS)}$
Recall	86.55	100	0.051	-21.10
Precision	92.24	54.46	0.047	64.59
Accuracy	88.61	54.46	0.048	57.77
F1	89.24	70.50	0.043	35.13

Table 5.9: Student's T-test results on STS and NLDS

Similarly, the two-sample unpaired t-test is applied to compare the performance between NLDS and the other baseline method RAE. A 5-fold cross validation is performed. 65 out of 325 instances are utilized as testing set, so the degree of freedom in this case is 128. The results are shown in table 5.10:

Tested Parameter	$\bar{X_{NLDS}}$**(%)**	$\bar{X_{RAE}}$**(%)**	$S_{X_{NLDS}X_{RAE}}$**(%)**	$t_{(NLDS,RAE)}$
Recall	86.55	100	0.051	-21.10
Precision	92.24	54.46	0.047	64.59
Accuracy	88.61	54.46	0.048	57.77
F1	89.24	70.50	0.043	35.13

Table 5.10: Student's T-test results on RAE and NLDS

According to the T-Distribution Table [1], we find that with a confidence factor $\alpha = 0.05$, $d = 128$, the threshold T is below 1.65. Table 5.9 and Table 5.10 show that the t value of our test is much larger than the threshold. Therefore, from the statistic perspective, with a 95% confidence degree, we claim that the performance of our NLDS is significantly better than STS and RAE, in terms of precision, accuracy, and F1-measure. Since both RAE and STS would recognize all the candidate instances as true cases, the recall ratio of NLDS is significantly lower than STS and RAE. However, this does not indicate the NLDS is defeated by STS and RAE. On the contrary, the tests reveal both STS and RAE are unable to distinguish non-uniform language instance from false candidates. In other words, the experiments demonstrate our NLDS is the most efficient solution to NLD task to date.

5.7 Discussion

As Table 5.7 shows, the PD system STS and RAE would regard all the candidate sentence pairs as true non-uniform language, so the recall ratio would be 1 but the precision would be very low.

It is worth noting that using character N-gram analysis only could not get good results. That is because the stream based analysis using probabilistic method is unable to capture any difference or relatedness in the meaning, while the NLD task relies heavily on discovering such relatedness. The reason we applied character N-gram analysis is to use it as a supplementary method to catch the difference such as between `shut down` (two words) and `shutdown` (one word), or some spelling errors.

POS analysis could provide a syntactic perspective for the instances. For the instances such as `then(/RB)` versus `and(/CC)`, and `store(/NN)` versus `stores(/NNS)`,

the difference could be reflected in POS tags. But still, POS analysis could not capture the difference between examples such as `writing(/VBG)` versus `entering(/VBG)` since they share the same POS tag. These features make POS analysis perform better than stream based analysis, but not as well as semantic analysis.

GTM achieves the best recall ratio when applied independently, since it could provide a semantic relatedness for the candidate instances, which is the most important feature for the NLD task. Resources such as WordNet and Flickr are utilized as supplementary resources to provide lexical relatedness and pragmatic relatedness for GTM.

By combining the different types of analysis above, the differences of each sentence pair are well analyzed at different NLP levels and thus, the relatedness and difference from structural, grammatical, syntactic, semantic and pragmatic perspectives would be captured and integrated by the classification method. Considering the different human judgments as reflected by Fleiss' Kappa Value, the NLD task is fairly difficult. The performance of our system is close to the human performance.

Chapter 6

Conclusion

This thesis proposes a solution to detect non-uniform language for technical writings at sentence level. Text, string-based, and word similarity algorithms at the lexical, syntactic, semantic, and pragmatic levels are integrated through an SVM based classification method. To evaluate the proposed method, we firstly invited three human experts to manually label all the candidate instances that were generated in stage 1. Then we assigned the ground truth for each instance pair by experts' voting. We then calculated the Fleiss's Kappa Value for each human expert to reflect the difference of their judgments and to reveal the difficulty of this task.

We also evaluated each human expert against the ground truth, and then regarded the results as the upper bound performance for this task. With the generated ground truth, a series of experiments using our implemented system were carried out with different smart phone user manual data. We evaluated the result by comparing it with the results using each single text similarity method as well as the state-of-the-art PD methods. The results reveal that our solution is the most effective method to date and the performance is close to the upper bound that we defined. As for future work, we would focus on performing deeper analysis on true non-uniform language pairs and try to provide suggestion to authors indicating which sentence within a pair is better, and then provide an automatic correction function.

Bibliography

[1] Student's t critical values. https://people.richland.edu/james/lecture/m170/tbl-t.html.

[2] Wordnet similarity demo. http://ws4jdemo.appspot.com/?mode=w&s1=&w1=photo&s2=&w2=video, 2015.

[3] Apple Inc. iPhone User Guide For iOS 8.4 Software. https://manuals.info.apple.com/MANUALS/1000/MA1565/en_US/iphone_user_guide.pdf, 2015.

[4] Regina Barzilay and Noemie Elhadad. Sentence alignment for monolingual comparable corpora. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 25–32. Association for Computational Linguistics, 2003.

[5] Regina Barzilay and Lillian Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. Association for Computational Linguistics, 2003.

[6] Thorsten Brants and Alex Franz. {Web 1T 5-gram Version 1}. https://catalog.ldc.upenn.edu/LDC2006T13, 2006.

[7] Thorsten Brants and Alex Franz. *Web 1T 5-gram Version 1*. Linguistic Data Consortium, 2006.

[8] Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8):1157–1166, 1997.

[9] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM, 2002.

[10] Francis Y. L. Chin and C. K. Poon. A fast algorithm for computing longest common subsequences of small alphabet size. *J. Inf. Process.*, 13(4):463–469, April 1991.

[11] Jacob Cohen. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.

[12] Paul R Cohen and Adele E Howe. How evaluation guides ai research: The message still counts more than the medium. *AI magazine*, 9(4):35, 1988.

[13] Joao Cordeiro, Gael Dias, and Pavel Brazdil. A metric for paraphrase detection. In *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*, pages 7–7. IEEE, 2007.

[14] Dipanjan Das and Noah A Smith. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 468–476. Association for Computational Linguistics, 2009.

[15] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[16] Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, page 350. Association for Computational Linguistics, 2004.

[17] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29(2-3):103–130, 1997.

[18] Joseph L Fleiss and Jacob Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 1973.

[19] Flickr. The home for all your photos. `https://www.flickr.com/`, 2015.

[20] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[21] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, volume 7, pages 1606–1611, 2007.

[22] Junbin B Gao, Steve R. Gunn, Chris J. Harris, and Martin Brown. A probabilistic framework for svm regression and error bar estimation. *Machine Learning*, 46(1-3):71–89, 2002.

[23] Wael H Gomaa and Aly A Fahmy. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13), 2013.

[24] Caichun Gong, Yulan Huang, Xueqi Cheng, and Shuo Bai. Detecting near-duplicates in large-scale short text databases. In *Advances in Knowledge Discovery and Data Mining*, pages 877–883. Springer, 2008.

[25] Patrick AV Hall and Geoff R Dowling. Approximate string matching. *ACM computing surveys (CSUR)*, 12(4):381–402, 1980.

[26] Vasileios Hatzivassiloglou, Judith L Klavans, and Eleazar Eskin. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proceedings of the 1999 joint sigdat conference on empirical methods in natural language processing and very large corpora*, pages 203–212. Citeseer, 1999.

[27] Robert W. Irving and Campbell Fraser. Two algorithms for the longest common subsequence of three (or more) strings. In *Proceedings of the Third Annual Symposium on Combinatorial Pattern Matching*, CPM '92, pages 214–229, London, UK, UK, 1992. Springer-Verlag.

[28] Aminul Islam and Diana Inkpen. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10, 2008.

[29] Aminul Islam, Evangelos Milios, and Vlado Kešelj. Text similarity using google tri-grams. In *Advances in Artificial Intelligence*, pages 312–317. Springer, 2012.

[30] Paul Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[31] Jay J Jiang and David W Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*, 1997.

[32] Vlado Kešelj and Nick Cercone. Cng method with weighted voting. In *Ad-hoc Authorship Attribution Competition. Proceedings 2004 Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities (ALLC/ACH 2004), Göteborg, Sweden*, 2004.

[33] Zoltán Kövecses and Günter Radden. Metonymy: Developing a cognitive linguistic view. *Cognitive Linguistics (includes Cognitive Linguistic Bibliography)*, 9(1):37–78, 1998.

[34] Pat Langley and Stephanie Sage. Induction of selective bayesian classifiers. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 399–406. Morgan Kaufmann Publishers Inc., 1994.

[35] Claudia Leacock and Martin Chodorow. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283, 1998.

[36] LG. LG600G User Guide. `https://www.tracfone.com/images/en/phones/TFLG600G/manual.pdf`, 2009.

[37] Dekang Lin. Extracting collocations from text corpora. In *First workshop on computational terminology*, pages 57–63. Citeseer, 1998.

[38] Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. Detecting near-duplicates for web crawling. In *Proceedings of the 16th international conference on World Wide Web*, pages 141–150. ACM, 2007.

[39] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780, 2006.

[40] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. Introduction to wordnet: An on-line lexical database*. *International journal of lexicography*, 3(4):235–244, 1990.

[41] Mohamed Ben Aouicha Mohamed Ali Hadj Taieb and Abdelmajid Ben Hamadou. Computing semantic relatedness using wikipedia features. 2013.

[42] NLTK Project. Nltk 3.0 documentation, 2009.

[43] The College of Information Sciences and The Pennsylvania State University Technology. CiteSeerX. `http://citeseerx.ist.psu.edu/`.

[44] OpenNLP. Apache opennlp developer documentation. `https://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html#tools.sentdetect.detection`, 2015.

[45] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*, 1995.

[46] Samsung. Samsung 010505d5 cell phone user manual. `http://cellphone.manualsonline.com/manuals/mfg/samsung/010505d5.html?p=53`, 2011.

[47] Shreyes Seshasai. *Efficient near duplicate document detection for specialized corpora*. PhD thesis, Massachusetts Institute of Technology, 2009.

[48] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[49] Guido F Smits and Elizabeth M Jordaan. Improved svm regression using mixtures of kernels. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 3, pages 2785–2790. IEEE, 2002.

[50] Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*. 2011.

[51] Michael Strube and Simone Paolo Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *AAAI*, volume 6, pages 1419–1424, 2006.

[52] Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

[53] Yifang Sun, Jianbin Qin, and Wei Wang. Near duplicate text detection using frequency-biased signatures. In *Web Information Systems Engineering–WISE 2013*, pages 277–291. Springer, 2013.

[54] Johan AK Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, Joos Vandewalle, JAK Suykens, and T Van Gestel. *Least squares support vector machines*, volume 4. World Scientific, 2002.

[55] Cornell University. arXiv. `https://arxiv.org`.

[56] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[57] Vapnik N Vladimir and V Vapnik. The nature of statistical learning theory, 1995.

[58] TU Wien. Package 'e1071'. `https://cran.r-project.org/web/packages/e1071/e1071.pdf`, 2015.

[59] Kyle Williams and C Lee Giles. Near duplicate detection in an academic digital library. In *Proceedings of the 2013 ACM symposium on Document engineering*, pages 91–94. ACM, 2013.

[60] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994.

[61] Eric Yeh, Daniel Ramage, Christopher D Manning, Eneko Agirre, and Aitor Soroa. Wikiwalk: random walks on wikipedia for semantic relatedness. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 41–49. Association for Computational Linguistics, 2009.

Appendix A

Data Set

All the data we utilized in this work are public online. We mix the smart phone user manuals of iPhone, Samsung, and LG.

The iPhone user manual is available at:
https://manuals.info.apple.com/MANUALS/1000/MA1565/en_US/iphone_user_guide.pdf

The Samsung user manual is available at:
http://cellphone.manualsonline.com/manuals/mfg/samsung/010505d5.html?p=53

The LG user manual is available at:
https://www.tracfone.com/images/en/phones/TFLG600G/manual.pdf

By converting the user manuals to txt format file, and upload to our stage 1 implementation at:
http://abidalrahman-2.cs.dal.ca/ULDWeb/ULD.jsp

The similar sentences above certain threshold would be captured and output by pairs. All these pairs make up the data set DS, which is utilized in this work. The generated data is also available at:
https://drive.google.com/open?id=0B2VIwNLCQnpGbDZfSE81VHotSm8

Appendix B

Code and Analyzed Data

All the codes that utilized in this work, including the sentence paring and feature classification model, are available at:
https://drive.google.com/open?id=0B2VIwNLCQnpGWDgwVFZBUy01YkU

All the intermediate data and experiment results are available at:
https://drive.google.com/open?id=0B2VIwNLCQnpGbDZfSE81VHotSm8

Appendix C

Baseline Methods

The STS system is available at:
http://cgm6.research.cs.dal.ca:8080/DalTextWebApp/WordWord.do

The RAE system is available at:
http://www.socher.org/index.php/Main/DynamicPoolingAndUnfoldingRecursiveAutoenc oders-sForParaphraseDetection

Appendix D

Cross Validation Results

D.1 Cross Validation Result of NLDS

	fold 1	**fold 2**	**fold 3**	**fold 4**	**fold 5**
TP	31	30	31	32	29
TN	27	28	27	29	24
FP	3	2	1	2	5
FN	4	5	6	2	7
Precision(%)	91.18	93.75	96.88	94.11	85.29
Recall(%)	88.57	85.74	83.78	94.11	80.56
Accuracy(%)	89.23	89.23	89.23	93.84	81.53
F1(%)	89.86	89.55	89.86	94.12	82.86

Table D.1: Cross validation results of NLDS

D.2 Cross Validation Result of STS

	fold 1	**fold 2**	**fold 3**	**fold 4**	**fold 5**
TP	35	35	37	34	36
TN	0	0	0	0	0
FP	30	30	28	31	29
FN	0	0	0	0	0
Precision(%)	53.85	53.85	56.92	52.31	55.38
Recall(%)	100	100	100	100	100
Accuracy(%)	53.85	53.85	56.92	52.31	55.38
F1(%)	70	70	72.55	68.69	71.29

Table D.2: Cross validation results of STS

D.3 Cross Validation Result of RAE

	fold 1	**fold 2**	**fold 3**	**fold 4**	**fold 5**
TP	35	35	37	34	36
TN	0	0	0	0	0
FP	30	30	28	31	29
FN	0	0	0	0	0
Precision(%)	53.85	53.85	56.92	52.31	55.38
Recall(%)	100	100	100	100	100
Accuracy(%)	53.85	53.85	56.92	52.31	55.38
F1(%)	70	70	72.55	68.69	71.29

Table D.3: Cross validation results of RAE

Appendix E

Fleiss' Kappa Test

The original data and calculation steps, as well as the final result of Fleiss' Kappa Test is available at:
https://drive.google.com/open?id=12f4gn6qJgk1GEU4jz2kmPFFuTPYupA6gyo3G3U2fisg