Non-uniform Language Detection in Technical Writing

Weibo Wang[1], **Abidalrahman Moh'd**[1], **Aminul Islam**[2]
Axel J. Soto[3], and **Evangelos E. Milios**[1]

[1]Faculty of Computer Science, Dalhousie University, Canada
{weibo,amohd,eem}@cs.dal.ca
[2]School of Computing and Informatics, University of Louisiana at Lafayette, USA
aminul@louisiana.edu
[3]School of Computer Science, University of Manchester, UK
axel.soto@manchester.ac.uk

Abstract

Technical writing in professional environments, such as user manual authoring, requires the use of uniform language. Non-uniform language detection is a novel task, which aims to guarantee the consistency for technical writing by detecting sentences in a document that are intended to have the same meaning within a similar context but use different words or writing style. This paper proposes an approach that utilizes text similarity algorithms at lexical, syntactic, semantic and pragmatic levels. Different features are extracted and integrated by applying a machine learning classification method. We tested our method using smart phone user manuals, and compared its performance against the state-of-the-art methods in a related area. The experiments demonstrate that our approach achieves the upper bound performance for this task.

1 Introduction

Technical writing, such as creating device operation manuals and user guide handbooks, is a special writing task that requires accurate text to describe a certain product or operation. To avoid ambiguity and bring accurate and straightforward understanding to readers, technical writing requires consistency in the use of terminology and uniform language (Farkas, 1985). 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 identifying non-uniform sentences. Such sentences are intended to have the same meaning or usage within a similar context but use different words or writing style. However, even though non-uniform sentences tend to have similar wording, similar sentence pairs do not necessarily indicate a non-uniform language instance. For example, here are four similar sentence pairs cited from the iPhone user manual (Apple Inc., 2015), where only two pairs are true non-uniform language instances:

(1) tap the screen to **show** the controls.
tap the screen to **display** the controls.

(2) tap the screen to **show** the controls.
tap the screen to **display** the controls.

(3) 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.

(4) 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 or phrases that the other sentence does not have. Each pattern could be a true or false non-uniform language instance depending on the content and context. The word '**show**' and '**display**' are synonyms in Example (1). Both sentences convey the same meaning, so they are an instance of non-uniform language. In Example (2), even though '**enter**' and '**write**' are not synonyms, since the two sentences describe the same

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1892–1900,
Austin, Texas, November 1-5, 2016.

operation, they should be considered as non-uniform language as well. In Example (3), even though the only different words between the sentences, '**photo**' and '**video**', are both media contents, because they are different objects, they should not be regarded as non-uniform language. In Example (4), it is a false candidate because each sentence mentions different functions. However, the two sentences are unequal in length, thus it is hard to know what the extra phrase '**wi-fi and**' should be compared against. Therefore, it is challenging to distinguish true and false occurrences of non-uniform cases based on text similarity algorithms only, and finer grained analyses need to be applied. To address the problem of NLD, this paper proposes a methodology for detecting non-uniform language within a technical document at the sentence level. A schematic diagram of our approach is shown in Figure 1.

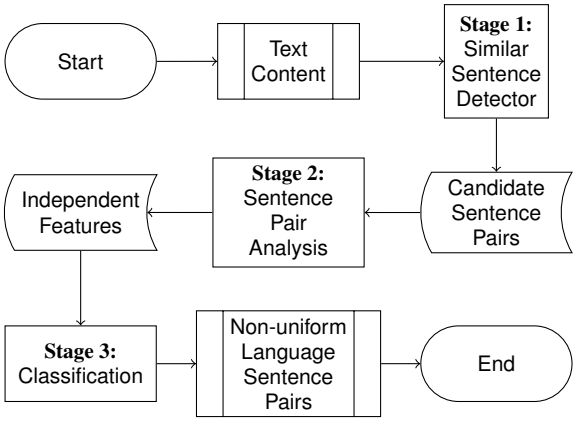

Figure 1: Schematic diagram of our approach

It is worth to mention that NLD is similar to Plagiarism Detection and Paraphrase Detection (PD) as all these tasks aim to capture similar sentences with the same meaning (Das and Smith, 2009). However, the goal of authors in plagiarism and paraphrasing is to change as many words as possible to increase the differences between texts, whereas in technical writing, the authors try to avoid such differences, but they do not always succeed and thus NLD solutions are needed. Cases of plagiarism and paraphrasing with high lexical differences will be typically classified as NLD negative, and cases with low lexical differences will be typically classified as NLD positive. While true positive cases for both NLD and PD can exist, there are not likely to happen in practice since textual differences in PD tend to be much higher than in NLD.

To address the NLD task, Natural Language Processing (NLP) techniques at lexical, syntactic, semantic, and pragmatic levels are utilized. Our approach also integrates resources such as Part-of-Speech (POS) tagger (Bird et al., 2009), WordNet (Miller et al., 1990), Google Tri-gram Method (GTM) (Islam et al., 2012; Mei et al., 2015), and Flickr[1]. Analyses from different perspectives are applied, and the results are regarded as independent features that are finally integrated by applying a classification method based on Support Vector Machine (SVM). A ground truth dataset created from three smart phone user manuals is used for evaluation, and it is made publicly available[2]. The experiments on this dataset demonstrate that the proposed solution to the NLD task is the most efficient method to date, and the final result is close to the upper bound performance.

2 Related Work

NLD is closely related to PD, which aims to detect sentences that have essentially the same meaning. However, paraphrase is a restatement using different words to make it appear different from the original text. PD techniques cannot perform well on the NLD task as they focus on variations at a coarser granularity. We reviewed studies in the PD area, and found the Recursive Auto-Encoder (RAE) (Socher et al., 2011), and the Semantic Text Similarity (STS) (Islam and Inkpen, 2008) to be the state-of-the-art methods using supervised and unsupervised-based PD, respectively. However, all the four examples provided in the introduction section would be recognized as paraphrases by these analyzers, even though only two of the pairs are real non-uniform language cases. Thus, state-of-the-art PD techniques are unable to make accurate judgments on these instances since PD do not address the necessary level of detail for the NLD task.

Another related area to NLD is near-duplicate text detection. It focuses on short text such as mobile phone short messages, or tweets, which are intended to have the same meaning but differ in terms of in-

[1]Flickr: https://www.flickr.com/

[2]The resource is available at: https://goo.gl/6wRchr

formal abbreviations, transliterations, and network languages (Gong et al., 2008). 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 (Sun et al., 2013). However, the studies in this area focus on reducing the comparison time in large scale text databases and creating informal abbreviation corpus, rather than exploring the text similarity methods. Basic similarity methods, such as Longest Common Substring (LCS) are utilized, but they are not sufficient to address the NLD task as LCS captures the matching words and their order between texts and using LCS alone will give high recall and low precision for the NLD task. For the following NLD negative example, LCS returns a high similarity score:

(5) If the **photo** hasn't been downloaded yet, tap the download notice first.
If the **music** hasn't been downloaded yet, tap the download notice first.

Examples of this type are common in technical writing, so other features are needed besides LCS to recognize NLD positives.

There is a research domain named near-duplicate document detection, which seems literally related to NLD, but also represents a different task. It focuses on documents that are identical in terms of written content but differ in a small portion of the document such as advertisements, counters and timestamps (Manku et al., 2007). Such documents are important to be identified for web crawling and the automatic collection of digital libraries. Since this area focuses on the variations between two documents, especially the variations on metadata, rather than the written content within one document, their proposed solutions are not a good fit for the NLD tasks.

3 Non-uniform Language Detection

As we have shown in Figure 1, a framework consisting of three stages is proposed to address the NLD task. The first stage extracts candidate sentence pairs that have high text similarity within a document. The second stage performs comprehensive analyses on each candidate sentence pair. The analyses are performed at lexical, syntactical, semantic, and pragmatic levels, where multiple NLP resources such as POS tagger, 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 cases of non-uniform language.

3.1 Stage 1: Similar Sentences Detection

To extract the candidate sentence pairs, three text similarity algorithms are combined and applied at the sentence level. GTM is an unsupervised corpus-based approach for measuring semantic relatedness between texts. LCS focuses on the word order of sentences. Cosine Similarity provides bag-of-word similarity. GTM, LCS, and Cosine Similarity are used to filter out the pairs based on semantics, sentence structure, and word frequency, respectively.

The filtering thresholds were set by running experiments at the sentence level on the iPhone user manual (Apple Inc., 2015). Algorithm 1 is used to set the filtering threshold for each average sentence length[3].

We utilize a sentence detector and a tokenizer[4] to divide the text of the manual into a sentence set of n sentence pairs (Line 2). We separately run Algorithm 1 three times to set the threshold sets for GTM, LCS, and Cosine. The thresholds are set based on the lengths of both sentences of a sentence pair. The average length starts from 2 and is increased by one once the threshold for the current length is set. We discovered that once the sentence length goes above 10, the thresholds vary little. Therefore, we stop the algorithm when the threshold for pairs of average length equal to 10 is found (Line 6).

For each different average length, the algorithm starts by asking the user to input an initial similarity threshold and an increasing step value (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 add the pairs within the current threshold range into set C (Line 14-16).

[3]See the Example (4) in Section 1, where two sentences within one sentence pair could be unequal in length, thus we compute the average length to represent the length of each candidate pair.

[4]OpenNLP: `https://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html`

Input : User Manual
Output: Threshold-Length_List $[(T_1, L_1), ...]$

1 **begin**
2 $S[n] \longleftarrow$ *SentenceDetector*(User Manual)
3 $L \longleftarrow 2$ /*Initial average length of a sentence pair*/
4 $T \longleftarrow$ Similarity threshold
5 $Step \longleftarrow$ Threshold increasing step
6 **while** $(L \leq 10)$ **do**
7 $C \longleftarrow \emptyset$ /*Initialize the output sentence container.*/
8 **do**
9 $T_{low} \longleftarrow T$
10 $T_{up} \longleftarrow T + Step$
11 **for** *(i=0; i<n; i++)* **do**
12 **for** *(j=0; j<n; j++)* **do**
13 $AvgL \longleftarrow (S[i] + S[j])/2$
14 **if** $AvgL \in [L-1, L)$ **then**
15 **if** $(T_{low} \leq Sim(S[i], S[j]))$ **and** $(Sim(S[i], S[j]) \leq T_{up})$ **then**
16 $C \overset{add}{\longleftarrow} (S[i], S[j])$
17 **end**
18 **end**
19 **end**
20 **end**
21 $T \longleftarrow T+Step$
22 **while** *(Check(C)=True)* /*Checked by human, True when all the sentence pairs are not instances of non-uniform language.*/;
24 *Threshold-Length_List* $\overset{add}{\longleftarrow} (T_{low}, L)$
25 $L++;$
26 **end**
27 **end**

Algorithm 1: Setting similarity thresholds

The similarity of sentence pairs above the previous threshold and below the current threshold are captured and analyzed (Line 15-16). If they consist of all false non-uniform language candidates, we repeat the loop with a higher threshold to filter more false candidates. Once we discover that a true candidate is filtered by the current thresholds, 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 2.

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 all the GTM, LCS and Cosine thresholds (which are 0.943, 0.836, and 0.932, according to Figure 2) to make it a candidate instance.

By applying the thresholds shown in Figure 2, candidate pairs could be detected in reasonable scale in terms of the size of the corpus, and achieve good

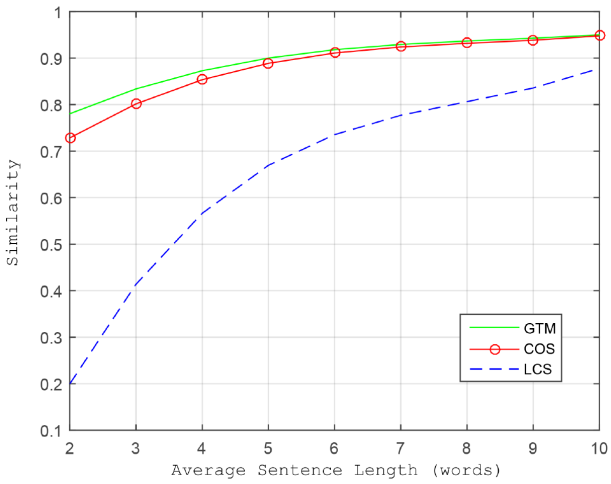

Figure 2: Candidate filtering thresholds

recall ratio as well. As for precision, around 40% of the candidates are true non-uniform language cases, where the remaining candidates are supposed to be filtered in the second stage.

3.2 Stage 2: Sentence Pair Analysis

In this stage, we aim to determine for the two sentences of a candidate pair whether they describe the same object or operation using different words or writing style (i.e., true non-uniform language) or they just appear similar but actually have different intended meanings, by using the following features.

3.2.1 Part-of-Speech Tagging Analysis

POS tags are added for each candidate pair using NLTK (Bird et al., 2009) tagger to gain a grammatical view over the sentences.

As Table 1 shows, some differences in sentence content can be captured using POS tags, but some cannot. Thus, it is necessary to make further syntactic and semantic analysis to distinguish true candidates from false ones.

We categorized the different POS tags into the following groups shown in Table 2. The different POS tags are mapped to different categories, which are then used as one more feature of the sentence pair representation.

3.2.2 Character N-gram Analysis

In the character N-gram analysis, the relatedness between the different words of each candidate pair is calculated in terms of character unigram, bigram

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

Table 1: POS analysis on candidate sentence pairs

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 2: POS tag categorizing

and trigram similarity. The character N-gram frequencies with a window size from 1 to 3 is firstly calculated. Then, the N-gram distance based on the frequencies is calculated using the Common N-gram distance (CNG) (Kešelj and Cercone, 2004):

$$d(f_1, f_2) = \sum_{n \in dom(f_1) \cup dom(f_2)} \left(\frac{f_1(n) - f_2(n)}{\frac{f_1(n)+f_2(n)}{2}}\right)^2 \quad (1)$$

where $dom(f_i)$ 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. The lower bound of the N-gram distance is 0 (when the two units to be compared are exactly the same). The higher the value of N-gram distance, the larger the difference, thus there is no upper bound. CNG was demonstrated to be a robust measure of dissimilarity for character N-grams in different domains (Wołkowicz and Kešelj, 2013).

3.2.3 WordNet Lexical Relation Analysis

For a given candidate sentence pair, if the different wordingF are synonymous to each other, there is a high likelihood that the two sentences try to convey the same meaning but using different expressions. 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 objects/actions and thus they might not be instances of non-uniform language.

WordNet is utilized here to analyze the lexical relationship within each candidate pair to determine whether they are synonyms to each other. To perform this analysis, we only used synset information from WordNet, and we only considered words as synonyms if they belong to a same synset. The rationale is that a similar sentence pair tends to be an instance of non-uniform language if 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. For example, given a similar sentence pair:

(6) 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 Wu-Palmer metric (Wu and Palmer, 1994) using WordNet is 0.6, which is fairly high compared to a random word pair. Yet we do not know how these words are related, e.g., "photo is a kind of video", "photo is a part of video", or "photo and video are examples of media content". Thus, we might make wrong judgments based on such a similarity score. However, using synset information, we know that these words are not synonyms and thus probably not suggesting a non-uniform language instance. Therefore, we considered as one more feature of our classifier whether mismatching words belong to the same synset.

3.2.4 GTM Word Relatedness Analysis

Besides text similarity, GTM also measures semantic relatedness between words. To find the relatedness between a pair of words, GTM takes into account all the trigrams that start and end with the given pair of words and then normalizes their mean frequency using unigram frequency of each of the words as well as the most frequent unigram in the Google Web 1T N-gram corpus (Brants and Franz, 2006), and extends the word relatedness method to measure document relatedness.

3.2.5 Flickr Related Concept Analysis

In some cases, word to word relatedness exists that goes beyond dictionary definitions, such 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 (Kövecses and Radden, 1998). Metonymy detection is actually a task at the pragmatic level of NLP area, which can be appied for NLD in technical writing.

Flickr 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 with the potential to find metonymy relationships in 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 obtained and regarded as another feature of our sentence pair representation for our NLD analysis. Table 3 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 3: Example of analysis using Flickr

3.3 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 different signals, a classification method based on SVM (Vladimir and Vapnik, 1995) is applied. We implemented the SVM classification using "e1071" package[5] in R.

Using our labeled corpus, we trained an SVM model on 61.5% of the data and used the remaining for testing.

4 Experiments and Evaluation

In this section, we present the dataset, experimental work and results, including results using other baseline methods for comparative purposes.

4.1 Experiment Data

We downloaded smart phone user manuals of iPhone (Apple Inc., 2015), LG (LG, 2009) and Samsung (Samsung, 2011), which are available online

[5]https://cran.r-project.org/web/packages/e1071/

as three raw datasets. Then, we performed Stage 1 three times on the three different datasets, and identified 325 candidate sentence pairs (650 sentences) as part of Stage 1, which is considered as our candidate dataset. Before applying the sentence analysis and classification stages, each candidate sentence pair in the dataset was labeled by three different annotators as true or false. Then the ground truth for each instance is generated by annotators' voting. The annotators worked separately to label the sentence pairs. Cases of disagreement were sent again to the annotators to double-check their judgement. Some statistics from the manuals are shown in Table 4.

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 4: Experiment data distribution

To prepare for the SVM based classification stage, we split the dataset into a training set DS_{train}, and a testing set DS_{test}. Considering that the data distribution is nearly balanced in terms of true and false instances, DS_{train} was formed by randomly selecting 200 instances from the dataset and the remaining 125 instances were used for DS_{test}.

4.2 Evaluation Methods and Results

The performance of each annotator against the majority voting is evaluated in terms of Precision, Recall, Accuracy, and F-measure. These results along with the number of true/ false, positive/ negative cases for each annotator are presented in Table 5.

Parameters	Expert 1	Expert 2	Expert 3
True-positive	130	99	125
True-negative	161	164	166
False-positive	20	51	25
False-negative	14	11	9
Precision	**86.67**	66.00	83.33
Recall	90.27	90.00	**93.28**
Accuracy	**89.54**	80.92	**89.54**
F-Measure	**88.43**	76.15	88.03

Table 5: Evaluation of annotators performance

To measure the agreement among annotators, the Fleiss' Kappa test (Fleiss and Cohen, 1973) is used. Fleiss' Kappa is an extension of Cohen's Kappa (Cohen, 1968). Unlike Cohen's Kappa, which only measures the agreement between two annotators, Fleiss' Kappa measures the agreement among three or more annotators. In our case, we have 3 annotators (the annotator number n is 3), each annotator 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). The final Fleiss' Kappa Value is 0.545, which indicates a moderate agreement level (0.41-0.60) based on the Kappa Interpretation Model (Fleiss and Cohen, 1973). In other words, the performance of the annotators reveal that the NLD task is not simple, since there are many cases that are ambiguous and hard to make accurate judgments on, even for humans.

As Table 5 shows, the best performance of annotators is highlighted and regarded as the upper bound performance (UB) of the NLD task on our dataset. The state-of-the-art unsupervised PD system named STS (Islam and Inkpen, 2008), as well as the state-of-the-art supervised PD system named RAE (Socher et al., 2011), are utilized to generate the baselines of the NLD task. STS uses the similarity score of 0.5 as the threshold to evaluate their method in the PD task. RAE applies supervised learning to classify a pair as a true or false instance of paraphrasing. These approaches are utilized on our evaluation as baselines for the NLD task.

After defining the upper bound and baseline performances, we evaluated our proposed method, which we name as Non-uniform Language Detecting System (NLDS), by training the SVM classifier on DS_{train}, and then performing classification using the SVM classifier on DS_{test}. The result is shown in Table 6 as the NLDS method. The first row presents 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 (Cohen and Howe, 1988) on N-gram, POS analysis, lexical analysis (GTM and WordNet), and Flickr, separately on the DS_{test} dataset. The results are listed in Table 6.

A series of cross-validation and Student's t-tests are applied after running NLDS, STS, RAE, and UB

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
Lexical	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 6: Evaluation of NLDS

methods on the F-measure metric. The tests reveal that the performance of NLDS is significantly better than STS and RAE, no significant differences could be found between UB and NLDS. These results demonstrate that NLDS would represent an effective approach for NLD that is on pair with annotator judgement and overcomes state-of-the-art approaches for related tasks.

4.3 Discussion

As Table 6 shows, the PD systems STS and RAE regard all the test cases as true non-uniform language cases, so the recall ratio is 1 but the precision is low.

It is worth noting that by using character N-gram analysis alone, it is not possible to obtain good results. This is because the character N-gram analysis using a 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 the N-gram analysis is to use it as a supplementary method to catch differences such as between '**shut down**' (two words) and '**shutdown**' (one word), or some spelling errors.

POS analysis provides a syntactic perspective for the text instances. For instances, '**then(/RB)**' versus '**and(/CC)**', and '**store(/NN)**' versus '**stores(/NNS)**', the differences can be reflected in POS tags. Yet, POS analysis alone could not capture the difference between words such as '**writing(/VBG)**' versus '**entering(/VBG)**' since they share the same POS tag. These features make POS analysis outperform the character N-gram analysis, but not semantic-based approaches.

Lexical analysis (GTM and WordNet) achieves the best recall ratio since it can provide semantic relatedness, which is the most important aspect for the NLD task. Flickr is utilized as a supplementary resource to provide pragmatic relatedness.

By combining the different types of analyses above, the differences of each sentence pair are analyzed at different NLP levels and thus, the relatedness and difference from structural, grammatical, syntactic, semantic and pragmatic perspectives can be captured and integrated by the classification method.

5 Conclusions

This paper proposes NLDS to detect non-uniform language for technical writings at sentence level. Text, stream-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, three annotators manually labeled all the candidate instances identified in Stage 1. Then we assigned the ground truth for each instance pair by annotators' voting. Fleiss' Kappa test is applied to reflect the difference of human judgments and thus to reveal the difficulty of this task.

We also evaluated each annotator against the ground truth, and defined the best performance of human 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 manuals data. We evaluated the results by comparing the outcome of the classifier with the results using each single feature, as well as the state-of-the-art PD methods.

Considering the different annotators' judgments as reflected by Fleiss' Kappa Value, the NLD task is fairly difficult. Yet, the performance of our system is close to human performance. The experiments 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 apply deeper analysis on true non-uniform language pairs to indicate which sentence of the pair fits better with the style and language of the rest of the document. We would then provide a semi-automatic correction function to facilitate authors with the task of removing non-uniform language occurrences.

Acknowledgments

This research work was supported by Innovatia Inc. and NSERC. We are thankful to our colleagues Andrew Albert, David Crowley, and Erika Allen who proposed and defined this NLD task, and provided expertise that contributed on the preparation of this paper.

References

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

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python*. " O'Reilly Media, Inc.".

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1.

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

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

Dipanjan Das and Noah A Smith. 2009. 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.

David K Farkas. 1985. The concept of consistency in writing and editing. *Journal of Technical Writing and Communication*, 15(4):353–364.

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

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

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

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

Vlado Kešelj and Nick Cercone. 2004. 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*.

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

LG. 2009. LG600G User Guide. https://www.tracfone.com/images/en/phones/TFLG600G/manual.pdf.

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

Jie Mei, Xinxin Kou, Zhimin Yao, Andrew Rau-Chaplin, Aminul Islam, Abidalrahman Moh'd, and Evangelos E. Milios. 2015. Efficient computation of co-occurrence based word relatedness. DemoURL: http://ares.research.cs.dal.ca/gtm/.

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

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

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

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

Vapnik N Vladimir and V Vapnik. 1995. The nature of statistical learning theory.

Jacek Wołkowicz and Vlado Kešelj. 2013. Evaluation of n-gram-based classification approaches on classical music corpora. In *Mathematics and computation in music*, pages 213–225. Springer.

Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. DemoURL:http://ws4jdemo.appspot.com/?mode=w&s1=&w1=photo&s2=&w2=video.