A Study of Incorrect Paraphrases in Crowdsourced User Utterances

Mohammad-Ali Yaghoub-Zadeh-Fard, Boualem Benatallah,
Moshe Chai Barukh and **Shayan Zamanirad**
University of New South Wales, Sydney
{m.yaghoubzadehfard,b.benatallah,mosheb,shayanz}@unsw.edu.au

Abstract

Developing bots demands high quality training samples, typically in the form of user utterances and their associated intents. Given the fuzzy nature of human language, such datasets ideally must cover all possible utterances of each single intent. Crowdsourcing has widely been used to collect such inclusive datasets by paraphrasing an initial utterance. However, the quality of this approach often suffers from various issues, particularly language errors produced by unqualified crowd workers. More so, since workers are tasked to write open-ended text, it is very challenging to automatically asses the quality of paraphrased utterances. In this paper, we investigate common crowdsourced paraphrasing issues, and propose an annotated dataset called *Para-Quality*, for detecting the quality issues. We also investigate existing tools and services to provide baselines for detecting each category of issues. In all, this work presents a data-driven view of incorrect paraphrases during the bot development process, and we pave the way towards automatic detection of unqualified paraphrases.

1 Introduction

With the increasing advances in deep learning as well as natural language processing, a new generation of conversational agents is attracting significant attention (Dale, 2016). Also known as *dialogue systems*, *virtual assistants*, *chatbots* or simply *bots* (Campagna et al., 2017; Su et al., 2017), some advanced bots are now designed to perform complex tasks (e.g., *flight booking*), many of which are built using machine learning techniques.

At the heart of building such *task-oriented bots* lies the challenge of accurately capturing the user's intent (e.g., *find cafes in Chicago*), and then extracting its entities to service the request (e.g *term= "cafes", location="Chicago"*). However, its success relies heavily on obtaining both, large and high quality corpora of training samples showing mappings between sample utterances and intents. This is necessary given the ambiguous nature of the human language (Wasow et al., 2005) and large variations of expressions (Wang et al., 2012; Zamanirad et al., 2017).

A lack of variations in training samples can result in incorrect intent detection and consequently execution of undesirable tasks (e.g., booking an expensive hotel instead of a cheap room) (Henderson et al., 2018). Likewise, quality issues in the training samples can lead to unmitigated disasters (Neff and Nagy, 2016) as it happened to Microsoft's *Tay* by making a huge number of offensive commentaries due to biases in the training data (Henderson et al., 2018). It is therefore not surprising that research and development into training data acquisition for bots has received significant consideration (Campagna et al., 2017; Kang et al., 2018).

Collecting training samples usually involves two primary steps: (i) firstly, obtaining an initial utterance for a given user intent (e.g., *find a cafe in Chicago*); and (ii) secondly, paraphrasing this initial expression into multiple variations (Su et al., 2017; Campagna et al., 2017). Paraphrasing is thus vital to cover the variety of ways an expression can be specified (Yang et al., 2018a). As summarized in (McCarthy et al., 2009), a quality paraphrases has three components: semantic completeness, lexical difference, and syntactic difference. To obtain lexically and syntactically diverse paraphrase, crowdsourcing paraphrases has gained popularity in recent years. However, crowdsourced paraphrases need to be checked for quality, given that they are produced by unknown workers with varied skills and motivations (Campagna et al., 2017; Daniel et al., 2018). For example, spammers, malicious and even inexperi-

Proceedings of NAACL-HLT 2019, pages 295–306
Minneapolis, Minnesota, June 2 - June 7, 2019.

enced crowd-workers may provide misleading, erroneous, and semantically invalid paraphrases (Li et al., 2016; Campagna et al., 2017). Quality issues may also stem from misunderstanding the intent or not covering important information such as values of the intent parameters (Su et al., 2017).

The common practice for quality assessment of crowdsourced paraphrases is to design another crowdsourcing task in which workers validate the output from others. However, this approach is costly having to pay for the task twice, making domain-independent automated techniques a very appealing alternative. Moreover, quality control is especially desirable if done before workers submit their paraphrases, since low quality workers can be removed early on without any payment. This can also allow crowdsourcing tasks to provide feedback to users in order to assist them in generating high quality paraphrases (Nilforoshan et al., 2017; Nilforoshan and Wu, 2018). To achieve this, it is therefore necessary to automatically recognize quality issues in crowdsourced paraphrases during the process of bot development.

In this paper, we investigate common paraphrasing errors when using crowdsourcing, and we propose an annotated dataset called *Para-Quality* in which each paraphrase is labelled with the error categories. Accordingly, this work presents a quantitative data-driven study of incorrect paraphrases in bot development process and paves the way towards *enhanced* automated detection of unqualified paraphrased utterances. More specifically, our contributions are two-folded:

- We obtained a sample set of 6000 paraphrases using crowdsourcing. To aim for a broad diversity of samples, the initial expressions were sourced from 40 expressions of highly popular APIs from various domains. Next, we examined and analyzed these samples in order to identify a taxonomy of common paraphrase errors errors (e.g., cheating, misspelling, linguistic errors). Accordingly, we constructed an annotated dataset called *Para-Quality* (using both crowdsourcing and manual verification), in which the paraphrases were labeled with a range of different categorized errors.

- We investigated existing tools and services (e.g., spell and grammar checkers, language identifiers) to detect potential errors. We formulated baselines for each category of errors to determine if they were capable to automatically detect such issues. Our experiments indicate that existing tools often have low precision and recall, and hence our results advocates the need for new approaches in effective detection of paraphrasing issues.

2 Paraphrase Dataset Collection

Various types of paraphrasing issues have been reported in the literature, namely: spelling errors (Braunger et al., 2018), grammatical errors (Jiang et al., 2017; Negri et al., 2012), and missing slot-value (happens when a worker forget to include an entity in paraphrases) (Su et al., 2017). We collected paraphrases for two main reasons: (i) to have a hands-on experience on how incorrect paraphrases are generated, and (ii) to annotate the dataset for building and evaluating paraphrasing quality control systems.

Methodology. We obtained 40 expressions from various domains (i.e. *Yelp, Skyscanner, Spotify, Scopus, Expedia, Open Weather, Amazon AWS, Gamil, Facebook, Bing Image Search*) indexed in ThingPedia (Campagna et al., 2017) and API-KG[1]. We then launched a paraphrasing task on Figure-Eight[2]. Workers were asked to provide three paraphrases for a given expression (Jiang et al., 2017), which is common practice in crowdsourced paraphrasing to reduce repetitive results (Campagna et al., 2017; Jiang et al., 2017). In the provided expression, parameter values were highlighted and crowd-workers were asked to preserve them. Each worker's paraphrases for an initial utterance are normalized by lowercasing and removing punctuation. Next, the initial utterance and the paraphrases are compared to forbid submitting empty strings or repeated paraphrases, and checked if they contain highlighted parameter values (which is also a common practice to avoid missing parameter values) (Mitchell et al., 2014). We collected paraphrases from workers in English speaking countries, and created a dataset containing 6000 paraphrases (2000 triple-paraphrases) in total[3].

3 Common Paraphrasing Issues

To characterize the types of paraphrasing issues, two authors of this paper investigated the crowd-

[1] http://apikg.ngrok.io
[2] https://www.figure-eight.com
[3] https://github.com/mysilver/ParaQuality

sourced paraphrases, and recognized 5 primary categories of paraphrasing issues. However, we only considered paraphrase-level issues related to the validity of a paraphrase without considering dataset-level quality issues such as lexical diversity (Negri et al., 2012) and bias (Henderson et al., 2018).

3.1 Spelling Errors

Misspelling has been reported as one of most common mistakes in paraphrasing (Inaba et al., 2015; Wang et al., 2012; Chklovski, 2005; Braunger et al., 2018). In our sample set, we also noticed misspellings were generated both intentionally (as an act of cheating to quickly generate a paraphrase such as Example 2 in Table 1) and unintentionally (due to a lack of knowledge or a simple mistake such as Example 3 in Table 1).

3.2 Linguistic Errors

Linguistic errors are also common in crowdsourced natural language collections (Jiang et al., 2017; Negri et al., 2012). Verb errors, preposition errors, vocabulary errors (improper word substitutions), and incorrect singular/plural nouns, just to name a few. Moreover, capitalization and article errors seems abundant (e.g., Example 5 in Table 1). Given that real bot users also make such errors, it is important to have linguistically incorrect utterances in the training samples (Bapat et al., 2018). However, at a very least, detecting linguistic errors can contribute to quality-aware selection of crowd workers.

3.3 Semantic Errors

This occurs when a paraphrase deviates from the meaning of the initial utterance (e.g., *find cafes in Chicago*). As reported in various studies, workers may forget to mention parameter values (also known as missing slot)(e.g., *find cafes*)[4] (Crossley et al., 2016; Su et al., 2017; Ravichander et al., 2017; Wang et al., 2012; Braunger et al., 2018), provide wrong values (e.g., *find cafes in Paris*) (Su et al., 2017; Ravichander et al., 2017; Wang et al., 2012; Negri et al., 2012; Braunger et al., 2018), or add unmentioned parameter values(Wang et al., 2012) (e.g., *find two cafes in Chicago*). Workers may also incorrectly use a singular noun instead of its plural form, and vice versa. For instance, in Example 6 of Table 1, the paraphrase only asks for the status of one specific *burglar alarm* while the expression asks for the status of all *burglar alarms*. Making mistakes in paraphrasing complementary forms of words also exists in the crowdsourced dataset. For instance, in Example 7 of Table 1, assuming that the bot answers the question only by saying "YES" or "NO", the answer for the paraphrase differs from that of the expression. However, it will make no difference if the bot's response is more descriptive (e.g., "it's working", "it isn't working".) Finally, some paraphrases significantly diverge from expressions. For instance, in Example 8 of Table 1, the intent of paraphrase is to turn off the TV; however, that of initial utterance is to query about the TV status.

[4] In our task design, this type of error cannot happen since parameter values are checked using regular expressions before submission

3.4 Task Misunderstanding

In some cases, workers misunderstood the task and provided translations in their own native languages (referred to as *Translation* issues) (Crossley et al., 2016; Braunger et al., 2018; Bapat et al., 2018), and some mistakenly thought they should provide answers for expressions phrased as questions (referred to as *Answering* issues) such as Example 9 in Table 1. This occurred even though workers were provided with comprehensive instructions and examples. We infer that some workers did not read the instructions, ignoring the possibility of cheating.

3.5 Cheating

In crowdsourced tasks, collecting paraphrases is not immune to unqualified workers, cheaters, or spammers (Daniel et al., 2018; Crossley et al., 2016; Chklovski, 2005).

Detecting malicious behaviour is vital because even constructive feedback may not guarantee quality improvements as workers act carelessly on purpose. *Cheating* is thus considered a special case of *Semantic Error* which is done intentionally. It is difficult even for experts to detect if someone is cheating or unintentionally making mistakes. However, it becomes easier when we consider all three paraphrases written by a worker for a given expression at once. For example, in Example 10 of Table 1, the malicious worker removes words one by one to generate new paraphrases. In this example, we also notice that it is still possible that a cheater produces a valid paraphrase accidentally such as the first paraphrase in Example 10. Workers may also start providing

#	Label		Sample
1	Correct	**Expression**	*Create a public playlist named new_playlist*
		Paraphrase	▷ Make a public playlist named new_playlist
2	Spelling Errors	**Expression**	*Estimate the taxi fare from the airport to home*
		Paraphrase	▷ Estimate the taxi fare from the airport to hom
3	Spelling Errors	**Expression**	*Estimate the taxi fare from the airport to home*
		Paraphrase	▷ Tell me about the far from airport to home
4	Spelling Errors	**Expression**	*Where should I try coffee near Newtown?*
		Paraphrase	▷ Find cafes near Newtown
5	Linguistic Errors	**Expression**	*Estimate the taxi fare from the airport to home*
		Paraphrase	▷ How much for taxi from airport to home
6	Semantic Errors	**Expression**	*Are the burglar alarms in the office malfunctioning?*
		Paraphrase	▷ Is the burglar alarm faulty in our work place?
7	Semantic Errors	**Expression**	*Are the burglar alarms in the office malfunctioning?*
		Paraphrase	▷ Are the office alarms working?
8	Semantic Errors	**Expression**	*Is the TV in the house off?*
		Paraphrase	▷ Can you turn off the TV in the house if it's on?
9	Task Misunderstanding	**Expression**	*Estimate the taxi fare from the airport to home?*
		Paraphrase	▷ Airport to home is $50
10	Cheating	**Expression**	*Request a taxi from airport to home*
		Paraphrases	▷ A taxi from airport to home ▷ Taxi from airport to home ▷ From airport to home
11	Cheating	**Expression**	*I want reviews for McDonald at Kensington st.*
		Paraphrases	▷ I want reviews g for McDonald at Kensington st. ▷ I want for reviews for McDonald at Kensington st. ▷ I want reviegws for McDonald at Kensington st.
12	Cheating	**Expression**	*I want reviews for McDonald at Kensington st.*
		Paraphrases	▷ I want to do reviews for McDonald's in Kensington st. ▷ I would like to do reviews for McDonald's in Kensington st. ▷ I could do reviews for McDonald's in Kensington st.
13	Cheating	**Expression**	*Create a public playlist named NewPlaylist*
		Paraphrases	▷ That song hits the public NewPlaylist this year ▷ Public really loved that NewPlaylist played on the event ▷ Public saw many NewPlaylist this year
14	Cheating	**Expression**	*Estimate the taxi fare from the airport to home*
		Paraphrases	▷ What is the fare of taxi from airport to home ▷ Tell me about fare from airport to home ▷ You have high taxi fare airport to home

Table 1: Paraphrase Samples

faulty paraphrases after generating some correct paraphrases as shown in Example 14 of Table 1. Based on our observations, the simplest mode of cheating is to add a few random characters to the source sentence as shown in Example 11. Next is adding a few words to the source sentence without much editing as shown in Example 12. Finally, there are cheaters who rewrite and change the sentences substantially in a very random way such as Example 13.

4 Dataset Annotation

Next, we designed another crowdsourcing task to annotate the collected paraphrases according to the category of issues devised above. Namely, using following labels: *Correct*, *Semantic Error*, *Misspelling*, *Linguistic Error*, *Translation*, *Answering*, and *Cheating*. We split the category of misunderstanding issues into *Translation* and *Answering* because they require different methods to detect.

Methodology. In the annotation task, crowd workers were instructed to label each paraphrase with the paraphrasing issues. Next, to further increase the quality of annotations [5], two authors of this paper manually re-annotated the paraphrases to resolve disagreements between crowd annotators. Moreover, contradictory labels (e.g., a paraphrase cannot be labeled both *Correct* and *Misspelling* simultaneously) were checked to ensure consistency. The overall Kappa test showed a high agreement coefficient between the annotators (McHugh, 2012) by Kappa being 0.85. Table 2 also shows the pair-wise inter-annotator agree-

[5] because of weak agreement between crowd workers

Label	Kappa
Correct	0.900
Misspelling	0.972
Linguistic Errors	0.879
Translation	1.000
Answering	0.855
Cheating	0.936
Semantic Errors	0.833

Table 2: Pairwise Inter-Annotator Agreement

ment (Cohen, 1960). Next, the authors discussed and revised the re-annotated labels to further increase the quality of annotations by discussing and resolving disagreements.

Statistics. Figure 1 shows the frequencies of each label in the crowdsourced paraphrases as well as their co-occurrences in an UpSet plot (Lex et al., 2014) using Intervene (Khan and Mathelier, 2017). Accordingly we infer that only 61% of paraphrases are labeled *Correct*. This plot also shows how many times two labels co-occurred. For example, all paraphrases which are labeled *Translation* (24 times), are also labeled *Cheating*[6].

5 Automatic Error Detection

Automatically detecting paraphrasing issues, especially when done during the crowd task, can minimize the cost of crowdsourcing by eliminating malicious workers, reducing the number of erroneous paraphrases, and eliminating the need for launching another crowdsourced validation task. Moreover, by detecting *Misspelling* and *Linguistic Errors*, users can be provided with proper feedback to help them improve the quality of paraphrasing by showing the source of error and suggestions to address the error (e.g., "*Spelling error detected: articl → article*"). Detecting *Semantic Errors*, such as missing parameter values, can also help crowd workers to generate high quality correct paraphrases. Automated methods can also be used to identify low quality workers, and particularly cheaters who may generate potentially large amount of invalid paraphrases intentionally. Moreover, providing suggestions to cheaters will not help and therefore early detection is of paramount.

[6]We used *Google Translate* to check whether they were proper translations or just random sentences in other languages

Spell Checker	Precision	Recall	F1
Aspell[8]	0.249	0.618	0.354
Hunspell[9]	0.249	0.619	0.355
MySpell[10]	0.249	0.619	0.355
Norvig[11]	0.488	0.655	0.559
Ginger[12]	0.540	0.719	0.616
Yandex[13]	0.571	0.752	0.650
Bing Spell Check[14]	0.612	**0.737**	0.669
LanguageTool[15]	**0.630**	0.727	**0.674**

Table 3: Comparison of Spell Checkers

In a pre-hoc quality control approach for crowdsourced paraphrases, the most important metric seems to be the *precision* of detecting invalid paraphrases (Nilforoshan et al., 2017). That is because the main aim of using such a quality control approach is rejecting invalid paraphrases without rejecting correct ones (Burrows et al., 2013). This is essential because rejecting correct paraphrases would be unfair and unproductive. For instance, sincere and trustful crowd workers might not get paid as a result of false-positives (incorrectly detected errors). On the other hand, having a high *recall* in detecting invalid paraphrases is important to eliminate faulty paraphrases and consequently obtain robust training samples.

Moreover, such a quality control technique should ideally be domain-independent, accessible, and easily-operated to minimize the cost of customization for a special domain and requiring paid experts (e.g., an open source pre-built machine learning model). In the rest of this section, we examine current tools and approaches and discuss their effectiveness in assessing the paraphrasing issues.

5.1 Spelling Errors

We employed several spell checkers as listed in Table 3 to examine if they are effective in recognizing spelling errors. We looked up Wikipedia, Github, and ProgrammableWeb[7] to find available tools and APIs for this purpose.

[7]https://www.programmableweb.com
[8]http://aspell.net/
[9]http://hunspell.github.io/
[10]http://www.openoffice.org/lingucomponent/dictionary.html
[11]https://github.com/barrust/pyspellchecker
[12]https://www.gingersoftware.com/ grammarcheck
[13]https://tech.yandex.ru/speller/
[14]https://azure.microsoft.com/en-us/services/cognitive-services/spell-check/
[15]https://languagetool.org

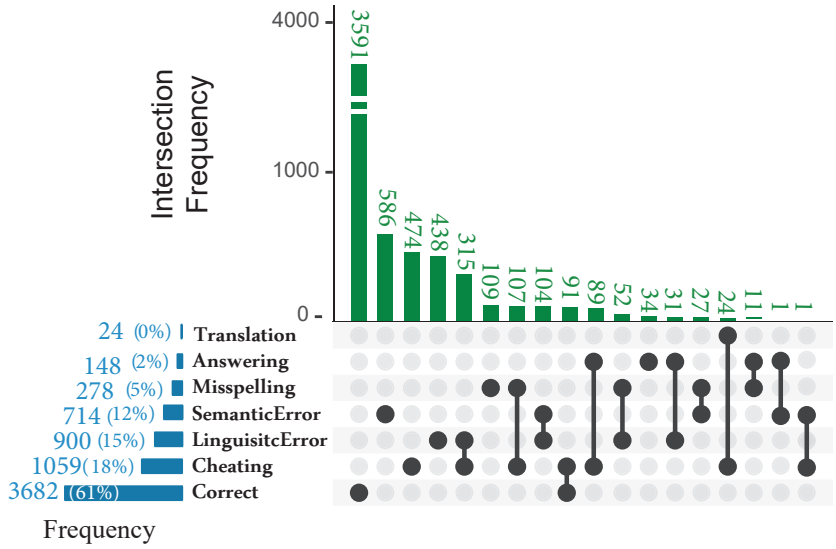

Figure 1: Dataset Label Statistics

Even though detecting misspelled words seems easy with existing automatic spellcheckers, they fall short in a few cases. This can be also concluded from Table 3 by considering the precision and recall of each spell checker in detecting only paraphrases with misspellings. For instance, spell checkers are often unable to identify homonyms (Perelman, 2016), incorrectly mark proper nouns and unusual words (Bernstein et al., 2015), and sometimes do not identify wrong words that are properly spelled (Chisholm and Henry, 2005). For instance, in Example 1 of Table 1, the "*new_playlist*" is incorrectly detected as a misspelled word by LanguageTool (the best performer as listed in Table 3). In Example 3, the word "*far*" is not detected even though the worker has misspelled the word "*fare*". In Example 4, the word "Newtown" (a suburb in Sydney) is mistakenly detected as a misspelling error. Some of these deficiencies can be addressed. For instance, in the case of spelling errors, assuming that the initial expressions given to the crowd are free of typos, we can ignore false-positives like the "Newtown" and "new_playlist".

5.2 Linguistic Errors

We investigated how well grammar checkers perform in detecting linguistic errors. We employed several grammar checkers as listed in Table 4.

Our experiments shows that spell checkers have both low precision and recall. Perelman (Perelman, 2016) also conducted several experiments with major commercial and non-commercial grammar checkers, and identified that

Grammar Checker	Precision	Recall	F1
AfterDeadline[17]	0.228	0.069	0.106
Ginger	0.322	**0.256**	**0.285**
GrammarBot[18]	0.356	0.139	0.200
LanguageTool	**0.388**	0.098	0.156

Table 4: Comparison of Grammar Checkers

grammar checkers are unreliable. Based on our observations, grammar checkers often fail in detecting linguistic errors as shown in Table 4. Examples include improper use of words (e.g., *"Who is the latest scientific article of machine learning?"*), random sequence of words generated by cheaters (e.g., *"Come the next sing"*), and missing articles[16] (e.g., *"I'm looking for flight for Tehran to Sydney"*). Given these examples, we believe that language models can be used to measure the likelihood of a sequence of words to detect if it is linguistically acceptable.

5.3 Translation

We also investigated several language detectors to evaluate how well they perform when crowd workers use another language instead of English. The results of experiment in Table 5 indicate that these tools detect almost all sentences in other languages. But they produce lots of false-positives including for correct English sentences (e.g., *"play next song"*). As a result, the tools in our experi-

[16]Missing articles in expressions similar to newspaper headlines are not considered error in the dataset (e.g., *"Hotel near Disneyland"*)

[17]https://www.afterthedeadline.com

[18]https://www.grammarbot.io

Language Detector	Precision	Recall	F1
FastText[19]	0.072	**1.000**	0.135
LangDetect[20]	0.080	0.917	0.147
LanguageIdentifier[21]	0.080	0.917	0.147
IBM Watson[22]	0.170	0.958	0.289
DetectLanguage[23]	0.344	0.917	0.500
DetectLanguage+	**0.909**	**1.000**	**0.952**

Table 5: Language Detection

ment have low precision in detecting languages as shown in Table 5. Most of the false-positives are caused by sentences that contain unusual words such as misspellings and named entities in the sentence (e.g., Email *Phil* saying "I got you").

One possible approach to improve the precision of such tools and APIs is to check if a given paraphrase has spelling errors prior to using language detection tools. We therefore extended the *DetectLanguage* (the best performing tool) by adding a constraint: a sentence is not written in another language unless it has at least two spelling errors. This constraint is based on the assumption that spell checkers treat foreign words as spelling errors and a sentence has at least two words to be called a sentence. This approach (*DetectLanguage+* in Table 5) significantly reduced the number of false-positives and thus improved precision.

5.4 Answering

Dialog Acts (DAs) (Jurafsky and Martin, 2018), also known as speech acts, represent general intents of an utterance. DA tagging systems label utterances with a predefined set of utterance types (*Directive, Commissive, Informative, etc* (Mezza et al., 2018).) Based on the fact that DAs must remain consistent during paraphrasing, we employed a state-of-art, domain-independent, pre-trained DA tagger proposed in (Mezza et al., 2018). For example, if an initial utterance is a *question* (e.g., *are there any cafes nearby?*) it is acceptable to paraphrase it into a *directive* sentence (e.g., *find cafes nearby.*), but its speech act cannot be *informative* (e.g., there is a cafe on the corner.). Overall, due to the lack of any other domain-independent DA tagger for the English language, we only investigated this tagger. We found that it has a precision of 2% with recall of 63%. This shows that detecting speech acts is a very challenging task especially for domain-independent environments.

Advances in speech act detection and availability of public speech act datasets can assist in detecting this category of the paraphrasing issues. Moreover, it is feasible to automatically generate pairs of questions and answers by mining datasets in the fields of Question Answering and dialog systems. Automatically building such pairs can help building a dataset which is diverse enough to be used in practice. Such a dataset can be fed into deep learning algorithms to yield better performance in detecting *Answering* issues.

5.5 Semantic Errors & Cheating

To the best our knowledge, there is not yet an approach to distinguish between categories of semantically invalid paraphrases. Paraphrase detection and textual semantic similarity (STS) methods are designed to measure how two pieces of text are semantically similar. However, they do not differentiate between different types of errors (e.g., *Cheating*, *Answering*, *Semantic Errors*) in our settings. As such, these techniques are not directly applicable. In the rest of this section, we focus on building machine learning models to detect the paraphrasing errors.

For this purpose, we used 38 established features from the literature as summarized in Table 6. Using these features and Weka (Hall et al., 2009), we built various classifiers to detect the following paraphrasing issues: *Answering*, *Semantic Errors*, and *Cheating*. We chose to test the five classification algorithms applied in paraphrasing literature as mentioned in (Burrows et al., 2013): C4.5 Decision Tree, K-Nearest Neighbor (K=50), Maximum Entropy, Naive Bayes, and Support Vector Machines (SVM) using default Weka 3.6.13 parameters for each of the classification algorithms. We also experimented with Random Forest algorithm since it is a widely-used classifier. We did not apply deep learning based classifiers directly due to the lack of expressions in the collected dataset which seems essential for developing domain independent classifiers. While our dataset is reasonably large, it contains only 40 expressions (each having 150 paraphrases). Given that deep learn-

[19] https://fasttext.cc/blog/2017/10/02/blog-post.html(Joulin et al., 2017)

[20] https://pypi.org/project/langdetect/

[21] https://github.com/saffsd/langid.py

[22] https://console.bluemix.net/apidocs/language-translator#identify-language

[23] https://ws.detectlanguage.com

Category	#	Description
N-gram Features	12	N-gram overlap, exclusive longest common prefix n-gram overlap, and SUMO all proposed in (Joao et al., 2007), as well as Gaussian, Parabolic, and Trigonometric proposed in (Cordeiro et al., 2007), Paraphrase In N-gram Changes (PINC) (Chen and Dolan, 2011), Bilingual Evaluation Understudy (BLEU) (Papineni et al., 2002), Google's BLEU (GLEU) (Wu et al., 2016), NIST (Doddington, 2002), Character n-gram F-score (CHRF) (Popović, 2016), and the length of the longest common subsequence.
Semantic Similarity	15	Semantic Textual Similarity (Fakouri-Kapourchali et al., 2018), Word Mover's Distance (Kusner et al., 2015) between words embeddings of expression and paraphrase, cosine similarity and euclidean distance between vectors of expression and paraphrase generated by Sent2Vec (Pagliardini et al., 2018), InferSent (Conneau et al., 2017), Universal Sentence Encoder (Cer et al., 2018), Concatenated Power Mean Embeddings (Rücklé et al., 2018), tenses of sentences, pronoun used in the paraphrase, and miss-matched named entities
Others	11	Number of spelling and grammatical errors detected by LanguageTool, task completion time, edit distance, normalized edit distance, word-level edit distance (Fakouri-Kapourchali et al., 2018), length difference between expression and paraphrase (in characters and words), and simple functions to detect questions, imperative sentences, and answering.

Table 6: Summary of Feature Library

Classifier	Precision	Recall	F1
Random Forest	**0.947**	0.129	0.226
Maximum Entropy	0.564	0.157	0.246
Decision Tree	0.527	**0.350**	**0.421**

Table 7: Automatic *Answering* Detection

Classifier	Precision	Recall	F1
Random Forest	**0.798**	0.120	0.209
Decision Tree	0.377	0.276	**0.319**
Naive Bayes	0.171	**0.783**	0.280

Table 8: Automatic *Semantic Error* Detection

Classifier	Precision	Recall	F1
SVM	**0.878**	0.223	0.356
K-Nearest Neighbor	0.871	0.248	0.386
Random Forest	0.843	0.546	**0.663**
Maximum Entropy	0.756	0.440	0.557
Decision Tree	0.632	**0.566**	0.597
Naive Bayes	0.473	0.426	0.449

Table 9: Automatic *Cheating* Detection

ing techniques are data thirsty (Goodfellow et al., 2016; Yang et al., 2018a), to use these kinds of models and eliminate the burden of manual feature engineering, much more expressions are needed. Instead, we benefited from the state-of-art sentence encoders via Transfer Learning as listed in Table 6.

Table 7, 8, and 9 demonstrate the performance of various classifiers (excluding classifiers with F1 being less than 0.2) for each of paraphrasing issues using 10-fold cross validation. To keep the classifiers domain-independent, we split the dataset based on the expressions without sharing any paraphrases of a single expression between the test and train samples. It can be seen that automatically detecting these quality issues is very challenging; even the best performing classifier has a very low F1 score especially for detecting *Answering* and *Semantic Error* issues. Based on manual exploration, we also found that the classifiers fail to recognize complex cheating behaviours such as Example 13 in Table 1 as discussed in Section 3. Therefore, new approaches are required to accurately detect paraphrasing issues. Based on our explorations and a prior work (McCarthy et al., 2009), we postulate that accurately detecting linguistic errors such as grammatically incorrect paraphrases can play indispensable role in detecting cheating behaviours. Moreover, advances in measuring semantic similarity between sentences can help differentiate between semantically invalid paraphrases and correct ones.

5.6 Incorrect Paraphrases Detection

We also assessed the performance of detecting incorrect paraphrases regardless of their categories. In this setting, we labeled all incorrect sentences with a single label (*"Incorrect"*) regardless of their categories. Table 10 demonstrates the performance of various classifiers. Detecting incorrect paraphrases is useful for post-hoc quality control to remove incorrect paraphrases after crowdsourcing paraphrases and consequently eliminate the need for crowdsourced validation task.

6 Related Work

To the best of our knowledge, that our work is the first to categorize paraphrasing issues and propose

Classifier	Precision	Recall	F1
K-Nearest Neighbor	**0.799**	0.341	0.478
Random Forest	0.781	**0.551**	**0.646**
Maximum Entropy	0.721	0.489	0.583
SVM	0.709	0.289	0.411
Decision Tree	0.633	0.585	0.608
Naive Bayes	0.574	0.557	0.565

Table 10: Automatic Incorrect Paraphrase Detection

an annotated dataset for assessing quality issues of paraphrased user expressions. Nevertheless, our work is related to the areas of (i) quality control in crowdsourced natural language datasets; and (ii) semantic similarity.

Quality Control. Quality can be assessed after or before data acquisition. While *post-hoc* methods evaluate quality when all paraphrases are collected, *pre-hoc* methods can prevent submission of low quality paraphrases during crowdsourcing. The most prevalent *post-hoc* approach is launching a verification task to evaluate crowdsourced paraphrases (Negri et al., 2012; Tschirsich and Hintz, 2013). However, automatically removing misspelled paraphrases (Wang et al., 2012) and discarding submissions from workers with low/high task completion time (Ma et al., 2017) are also applied in literature. Machine learning models have also been explored in plagiarism detection systems to assure quality of crowdsourced paraphrases (Crossley et al., 2016; Burrows et al., 2013).

Pre-hoc methods, on the other hand, rely on online approaches to asses the quality of the data provided during crowdsourcing (Nilforoshan et al., 2017). Sophisticated techniques are required to avoid generation of erroneous paraphrases (e.g., automatic feedback generation was used to assist crowd workers in generating high quality paraphrases). Precog (Nilforoshan et al., 2017) is an example of such tools which is based on a supervised method for generating automatic writing feedback for multi-paragraph text– designed mostly for crowdsourced product reviews (Nilforoshan et al., 2017; Nilforoshan and Wu, 2018). This paper aims for paving the way for building automatic *pre-hoc* approaches, and providing appropriate online feedback to users to assist them in generating appropriate paraphrases. However, the provided dataset can also be used for building *post-hoc* methods to automatically omit faulty paraphrases.

Semantic Similarity. Measuring similarity between units of text plays an important role in Natural Language Processing (NLP). Several NLP tasks have been designed to cover various aspects and usages of textual similarity. Examples include textual entailment, semantic textual similarity (Yang et al., 2018b; Fakouri-Kapourchali et al., 2018), paraphrase detection (Agarwal et al., 2018; Issa et al., 2018), duplicate question detection (Mannarswamy and Chidambaram, 2018) tasks which are studied well in NLP. Moreover, recent success in sentence encoders (e.g., Sent2Vec (Pagliardini et al., 2018), InferSent (Conneau et al., 2017), Universal Sentence Encoder (Cer et al., 2018), and Concatenated Power Mean Embeddings (Rücklé et al., 2018)) can be exploited to detect paraphrasing issues with more accuracy. These techniques can be borrowed with some domain specific considerations to build automatic quality control systems for detecting low quality paraphrases.

7 Conclusion

In this paper, we employed a data-driven approach to investigate and quantitatively study various crowdsourced paraphrasing issues. We discussed how automatic techniques for detecting various quality issues can assist the manual process of crowdsourced paraphrasing. We collected an annotated dataset of crowdsourced paraphrasing in which each paraphrase is labeled with associated paraphrasing issues. We used this dataset to assess existing tools and techniques and to determine whether they are sufficient for automatically detecting such issues. Our experiments revealed that automated detection of errors in paraphrases is a challenging task. As a future work, we will be working on devising automated-assisted methods for detection of paraphrasing issues. This will be based on a two-way feedback mechanism: generating feedback for workers, while at the same time the system learns from the (data of) users to improve its machine intelligence. In time, we envision increasingly less dependence on users.

acknowledgements

This research was supported fully by the Australian Government through the Australian Research Council's Discovery Projects funding scheme (project DP1601104515).

References

Basant Agarwal, Heri Ramampiaro, Helge Langseth, and Massimiliano Ruocco. 2018. A deep network model for paraphrase detection in short text messages. *Information Processing & Management*, 54(6):922–937.

Rucha Bapat, Pavel Kucherbaev, and Alessandro Bozzon. 2018. Effective crowdsourced generation of training data for chatbots natural language understanding. In *Web Engineering*, Cham. Springer International Publishing.

Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. 2015. Soylent: A word processor with a crowd inside. volume 58, pages 85–94, New York, NY, USA. ACM.

Patricia Braunger, Wolfgang Maier, Jan Wessling, and Maria Schmidt. 2018. Towards an automatic assessment of crowdsourced data for nlu. In *LREC*.

Steven Burrows, Martin Potthast, and Benno Stein. 2013. Paraphrase acquisition via crowdsourcing and machine learning. volume 4, pages 43:1–43:21, New York, NY, USA. ACM.

Giovanni Campagna, Rakesh Ramesh, Silei Xu, Michael Fischer, and Monica S. Lam. 2017. Almond: The architecture of an open, crowdsourced, privacy-preserving, programmable virtual assistant. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 341–350, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

David L. Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 190–200, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wendy A. Chisholm and Shawn Lawton Henry. 2005. Interdependent components of web accessibility. In *Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*, W4A '05, pages 31–37, New York, NY, USA. ACM.

Timothy Chklovski. 2005. Collecting paraphrase corpora from volunteer contributors. In *Proceedings of the 3rd International Conference on Knowledge Capture*, K-CAP '05, pages 115–120, New York, NY, USA. ACM.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

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

Scott Crossley, Luc Paquette, Mihai Dascalu, Danielle S. McNamara, and Ryan S. Baker. 2016. Combining click-stream data with nlp tools to better understand mooc completion. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, LAK '16, pages 6–14, New York, NY, USA. ACM.

Robert Dale. 2016. The return of the chatbots. *Natural Language Engineering*, 22(5):811–817.

Florian Daniel, Pavel Kucherbaev, Cinzia Cappiello, Boualem Benatallah, and Mohammad Allahbakhsh. 2018. Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions. volume 51, pages 7:1–7:40, New York, NY, USA. ACM.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Roghayeh Fakouri-Kapourchali, Mohammad-Ali Yaghoub-Zadeh-Fard, and Mehdi Khalili. 2018. Semantic textual similarity as a service. In *Service Research and Innovation*, pages 203–215, Cham. Springer International Publishing.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*, volume 1. MIT press Cambridge.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. volume 11, pages 10–18, New York, NY, USA. ACM.

Peter Henderson, Koustuv Sinha, Nicolas Angelard-Gontier, Nan Rosemary Ke, Genevieve Fried, Ryan Lowe, and Joelle Pineau. 2018. Ethical challenges in data-driven dialogue systems. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and*

Society, AIES '18, pages 123–129, New York, NY, USA. ACM.

Michimasa Inaba, Naoyuki Iwata, Fujio Toriumi, Takatsugu Hirayama, Yu Enokibori, Kenichi Takahashi, and Kenji Mase. 2015. Statistical response method and learning data acquisition using gamified crowdsourcing for a non-task-oriented dialogue agent. In *Revised Selected Papers of the 6th International Conference on Agents and Artificial Intelligence - Volume 8946*, ICAART 2014, pages 119–136, Berlin, Heidelberg. Springer-Verlag.

Fuad Issa, Marco Damonte, Shay B Cohen, Xiaohui Yan, and Yi Chang. 2018. Abstract meaning representation for paraphrase detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 442–452.

Youxuan Jiang, Jonathan K. Kummerfeld, and Walter S. Lasecki. 2017. Understanding task design trade-offs in crowdsourced paraphrase collection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 103–109. Association for Computational Linguistics.

Cordeiro Joao, Dias Gaël, and Brazdil Pavel. 2007. New functions for unsupervised asymmetrical paraphrase detection. *Journal of Software*, 2(4):12–23.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

D Jurafsky and JH Martin. 2018. Dialog systems and chatbots. *Speech and language processing*.

Yiping Kang, Yunqi Zhang, Jonathan K Kummerfeld, Lingjia Tang, and Jason Mars. 2018. Data collection for dialogue system: A startup perspective. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, volume 3, pages 33–40.

Aziz Khan and Anthony Mathelier. 2017. Intervene: a tool for intersection and visualization of multiple gene or genomic region sets. *BMC bioinformatics*, 18(1):287.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*, pages 957–966.

Alexander Lex, Nils Gehlenborg, Hendrik Strobelt, Romain Vuillemot, and Hanspeter Pfister. 2014. Upset: visualization of intersecting sets. *IEEE transactions on visualization and computer graphics*, 20(12):1983–1992.

Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael J Franklin. 2016. Crowdsourced data management: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2296–2319.

Xiao Ma, Trishala Neeraj, and Mor Naaman. 2017. A computational approach to perceived trustworthiness of airbnb host profiles.

Sandya Mannarswamy and Saravanan Chidambaram. 2018. Geminio: Finding duplicates in a question haystack. In *Advances in Knowledge Discovery and Data Mining*, pages 104–114, Cham. Springer International Publishing.

Philip M. McCarthy, Rebekah H. Guess, and Danielle S. McNamara. 2009. The components of paraphrase evaluations. *Behavior Research Methods*, 41(3):682–690.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica*, 22(3):276–282.

Stefano Mezza, Alessandra Cervone, Evgeny Stepanov, Giuliano Tortoreto, and Giuseppe Riccardi. 2018. Iso-standard domain-independent dialogue act tagging for conversational agents. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3539–3551. Association for Computational Linguistics.

Margaret Mitchell, Dan Bohus, and Ece Kamar. 2014. Crowdsourcing language generation templates for dialogue systems. *Proceedings of the INLG and SIGDIAL 2014 Joint Session*, pages 172–180.

Gina Neff and Peter Nagy. 2016. Automation, algorithms, and politics— talking to bots: symbiotic agency and the case of tay. *International Journal of Communication*, 10:17.

Matteo Negri, Yashar Mehdad, Alessandro Marchetti, Danilo Giampiccolo, and Luisa Bentivogli. 2012. Chinese whispers: Cooperative paraphrase acquisition. In *LREC*, pages 2659–2665.

Hamed Nilforoshan, Jiannan Wang, and Eugene Wu. 2017. Precog: Improving crowdsourced data quality before acquisition. *CoRR*, abs/1704.02384.

Hamed Nilforoshan and Eugene Wu. 2018. Leveraging quality prediction models for automatic writing feedback. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, Stanford, California, USA, June 25-28, 2018.*, pages 211–220.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Les Perelman. 2016. Grammar checkers do not work. *WLN: A Journal of Writing Center Scholarship*, 40(7-8):11–20.

Maja Popović. 2016. chrf deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504. Association for Computational Linguistics.

Abhilasha Ravichander, Thomas Manzini, Matthias Grabmair, Graham Neubig, Jonathan Francis, and Eric Nyberg. 2017. How would you say it? eliciting lexically diverse dialogue for supervised semantic parsing. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 374–383.

Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. 2018. Concatenated p-mean word embeddings as universal cross-lingual sentence representations. *CoRR*, abs/1803.01400.

Yu Su, Ahmed Hassan Awadallah, Madian Khabsa, Patrick Pantel, Michael Gamon, and Mark Encarnacion. 2017. Building natural language interfaces to web apis. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 177–186, New York, NY, USA. ACM.

Martin Tschirsich and Gerold Hintz. 2013. Leveraging crowdsourcing for paraphrase recognition. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 205–213.

William Yang Wang, Dan Bohus, Ece Kamar, and Eric Horvitz. 2012. Crowdsourcing the acquisition of natural language corpora: Methods and observations. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pages 73–78. IEEE.

Thomas Wasow, Amy Perfors, and David Beaver. 2005. The puzzle of ambiguity. *Morphology and the web of grammar: Essays in memory of Steven G. Lapointe*, pages 265–282.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Jie Yang, Thomas Drake, Andreas Damianou, and Yoelle Maarek. 2018a. Leveraging crowdsourcing data for deep active learning an application: Learning intents in alexa. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 23–32, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-Yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2018b. Learning semantic textual similarity from conversations. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 164–174. Association for Computational Linguistics.

Shayan Zamanirad, Boualem Benatallah, Moshe Chai Barukh, Fabio Casati, and Carlos Rodriguez. 2017. Programming bots by synthesizing natural language expressions into api invocations. In *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*, pages 832–837. IEEE Press.