Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2023"

June 14–16, 2023

Augmentation methods for spelling corruptions

Nikita Martynov
SberDevices
Moscow
nikita.martynov.98@list.ru

Mark Baushenko
SberDevices
Moscow
MABaushenko@sberbank.ru

Alexander Abramov
SberDevices
Moscow
Abramov.A.Sergee@sberbank.ru

Alena Fenogenova
SberDevices
Moscow
alenush93@gmail.com

Abstract

The problem of automatic spelling correction is vital to applications such as search engines, chatbots, spell-checking in browsers and text editors. The investigation of spell-checking problems can be divided into several parts: error detection, emulation of the error distribution on the new data for model training, and automatic spelling correction. As the data augmentation technique, the adversarial training via error distribution emulation increases a model's generalization capabilities; it can address many other challenges: from overcoming a limited amount of training data to regularizing the training objectives of the models. In this work, we propose a novel multi-domain dataset for spelling correction. On this basis, we provide a comparative study of augmentation methods that can be used to emulate the automatic error distribution. We also compare the distribution of the single-domain dataset with the errors from the multi-domain and present a tool that can emulate human misspellings.

Keywords: spelling correction, augmentation strategies, adversarial attacks, error detection
DOI: 10.28995/2075-7182-2023-22-327-349

Методы аугментации для задачи автоматического исправления орфографии

Никита Мартынов
SberDevices
Moscow
nikita.martynov.98@list.ru

Марк Баушенко
SberDevices
Moscow
MABaushenko@sberbank.ru

Александр Абрамов
SberDevices
Moscow
Abramov.A.Sergee@sberbank.ru

Алена Феногенова
SberDevices
Moscow
alenush93@gmail.com

Аннотация

Автоматическая коррекция орфографии актуальна для многих приложений, таких как поисковые системы, чат-боты, текстовых редакторах и тд. Системы автоматического распознавания и исправления опечаток часто используют в кач-ве метода аугментации данных. Это повышает метрики оценки на низкоуровневых задачах, увеличивает обобщающую способность модели и её робастность. В этой работе мы впервые представляем новый многодоменный набор данных для исправления орфографии. На его основе мы предлагаем несколько подходов к аугментации данных и проводим сравнительную оценку методов увеличения данных с различными распределениями ошибок, которые можно в дальнейшем использовать для эмуляции автоматического распределения ошибок.

Ключевые слова: проверка орфографии, автоматическое определение ошибок, методы аугментации данных

1 Introduction

The task of automatic spelling correction (or spell-checking) is crucial for many applications in different areas, including correction of search queries, spell checking in browsers, text editors etc. There are plenty of methods for spelling detection and correction. In recent research, with new big language models, the generation of texts without spelling errors expands new horizons. There are various methods of automatic text corruption and augmentations for further model training on parallel texts. However, more reliable information on human error distribution in the text data needs to be found. How well existing approaches can approximate the natural error distribution is still an open question. The influence on the quality of the generative models trained on such data is also a new field for investigation.

In this paper, we deal with several of these research problems. Due to the lack of data for the Russian language of different domains with spelling errors, we present a new parallel dataset for spelling correction. We propose two methods for spelling correction. On this basis, we conduct a comparative study of these augmentation algorithms that can be used to emulate spelling error distribution. Our key contributions to the paper are the following:

- **We introduce a novel multi-domain dataset for spelling correction**. We compare the public single-domain dataset from the Shared Task SpellRuEval-2016 (Sorokin et al.,) with the obtained golden multi-domain set and prove that the domain distributions differ in various domains.
- **We propose two approaches to generate spelling error distribution.**
  - We introduce the augmentation method that emulates human spelling errors based on statistical data and heuristics from keyboard usage. Such a method can produce corrupted text without any labelled data. The obtained spelling error distribution from texts corrupted with this method is compared with the golden test sets spelling error distribution.
  - We provide the augmentation tool based on the method that gathers the error distribution from the parallel corpus and can replicate the obtained source distribution on a new text based on classic Levenshtein operations (Lhoussain et al., 2015) (deletion, insertion, substitutions). We clone the error distribution from the golden set and compare the emulated with the original distribution.

The remainder is structured as follows. First, we overview the approaches to spell correction 2, the available datasets and methods for error augmentations. Section 3 describes the data sources and the annotation procedure for creating the Russian multi-domain corpus. In section 4, we observe the augmentation methods and models used and provide the description of the comparable experiments. The statistical evaluation is presented in Section 5.

2 Related Work

The problem of spelling correction has a long history of research. It attracted intensive attention in the early era of modern NLP. The most significant early works are the edit distance model, introduced by Levenshtein (Levenshtein and others, 1966) and further by Damerau (Damerau, 1964). Weighted variants of error distances were considered in (Kemighan et al., 1990) and Brill and Moore (Brill and Moore, 2000). The latest also proposed the noisy channel error correction model based on n-grams. Toutanova and Moore (Toutanova and Moore, 2002) added a pronunciation model for spelling correction. A broad historical overview of the problem is presented in the paper (Shavrina, 2017), where the author discusses the history of methods of automatic spelling correction and the requirements faced by the systems implementing such methods at different historical stages.

The interest in this field for the Russian language appears after SpellRuEval-2016 (Sorokin et al.,) competition. The authors created the single domain dataset for social media texts and provided the first benchmark and standard for spelling correction problems. Among other public popular solutions for Russian language are Yandex.Speller [1], DeepPavlov [2] method based on Damerau Levenshtein and

[1] https://yandex.ru/dev/speller/

[2] https://docs.deeppavlov.ai/en/master/features/models/spelling_correction.html

KenLM, Hunspell [3], Jamspell [4]. It's necessary to mention a multilingual source of parallel spell data – GitHub Typo Corpus (Hagiwara and Mita, 2019). It is a large-scale, multilingual dataset of misspellings and grammatical errors along with their corrections harvested from GitHub. For state-of-the-art spelling systems, the generative models [5] are applied. For its training, the parallel corpus needs to be built from scratch; emulating spelling errors or augmentation of the existing datasets is required.

The approaches for error augmentations are common and applied in further research. For example, they are incorporated in the GEM benchmark (Dhole et al., 2021), and its augmentation NL-Augmenter library [6]. The (Benes and Burget, 2020) examines the effect of data augmentation for training language models for speech recognition and investigates the behaviour of perplexity estimated on augmented data. For the Russian Language frameworks RuTransform (Taktasheva et al., 2022) [7] adds noise to data via spelling corruption. It contains the ButterFingers method, employed at the word level, as well as the sentence-level techniques of word swapping (EDA_{SWAP}) and token deletion (EDA_{DELETE}). The ButterFingers method, derived from the NL-Augmenter, constitutes a typo-based perturbation approach that adds noise into textual data and Case methos introduces noise to data through case alteration. This is accomplished by simulating spelling errors made by humans through character swaps, taking into account the keyboard distance between the characters. Notably, these methods are applicable to both the Russian and English languages.

3 Data

We acquire text data from publicly available sources out of five domains to create a multi-domain corpus. Due to human and time constraints, all the texts are automatically checked for the presence of spelling mistakes. For the sentences with potential misspellings, we set up a two-stage human annotation procedure. As a result, we select 1711 parallel sentences based on the agreement between annotators. You can see the full breakdown in Table 1.

3.1 Data sources

The choice of the domains of our primary interest lays upon the following criteria:

- The texts from a particular domain must be misspellings-prone.
- The representation of a domain should reflect the frequency of misspellings present within it. By assuming that texts belonging to a particular domain inherently contain spelling errors, it follows that a larger corpus of texts would naturally yield a greater number of sentences, thus expanding the dataset.
- Finally, the resulting domains must be diverse in terms of vocabulary, grammatical structures, slang, jargon etc. It ensures we capture different types, positions and co-occurrences of misspellings.

These conditions lead to the following choice of domains and corresponding datasets.

Aranea web-corpus (Benko, 2014) is a family of multilanguage gigaword web-corpora collected from Internet resources. The texts in the corpora are evenly distributed across periods, writing styles and topics they cover. We randomly picked the sentences from Araneum Russicum [8], which is harvested from the Russian part of the web.

Literature is a collection of Russian poems and prose of different classical literary works. We randomly picked sentences from the source dataset[9] that were gathered from Ilibrary, LitLib, and Wikisource.

News, as the name suggests, covers news articles on various topics such as sports, politics, environment, economy etc. The passages are randomly picked from the summarization dataset Gazeta.ru. (Gusev, 2020)

[3] https://github.com/pyhunspell/pyhunspell
[4] https://github.com/bakwc/JamSpell
[5] https://huggingface.co/UrukHan/t5-russian-spell
[6] https://github.com/GEM-benchmark/NL-Augmenter
[7] https://github.com/RussianNLP/rutransform
[8] http://ucts.uniba.sk/aranea_about/_russicum.html
[9] https://www.kaggle.com/datasets/d0rj3228/russian-literature

Social media is the text domain from social media platforms marked with specific hashtags. These texts are typically short, written in an informal style and may contain slang, emojis and obscene lexis.
Strategic Documents is part of the dataset the Ministry of Economic Development of the Russian Federation collected. Texts are written in a bureaucratic manner, rich in embedded entities, and have complex syntactic and discourse structures. The full version of the dataset has been previously used in the RuREBus shared task (Ivanin et al., 2020).

Datasets	Raw texts	Yandex.Speller	Filtered texts	First stage	Second stage
Aranea web-corpus	45512	3761	985	859	756
Literature	24635	1808	494	262	260
News	2001	245	245	245	245
Social media	25883	3000	281	208	200
Strategic Documents	44458	2000	284	250	250
TOTAL	**142489**	**10814**	**2289**	**1824**	**1711**

Table 1: The number of sentences on all stages of the dataset creation among all domains. *Raw texts* is several texts in the source; *Yandex.Speller* is a number of texts marked by Yandex.Speller that can contain misspellings. *Filtered texts* reflects texts sent to manual labeling; *First stage* corresponds to the texts passed to second stage of labeling; *Second stage* is a number of resulting sentences.

3.2 Candidate selection

First, we automatically detect mistakes with Yandex.Speller[10]. We find out that Yandex.Speller is often triggered by proper names, slang, abbreviations, obsolete and rare word forms (see Table 2 for illustrative examples) that do not contain any spelling errors.

Second, in this paper we do not consider specific vocabulary, e.g. slang, jargonisms, colloquialisms etc., as an error, as we see them as style markers that reflect distinctive domain features. For example, the word *"ето"* in a sentence *"тут ето, коты синхронизировались"* *("here it is, the cats are synchronized")* from Social media domain is not correct in terms of a standard language. Still, this word is presumably used to endow a sentence with a particular emotional expression. Nevertheless, we do not allow all the misspellings in specific vocabulary - we only keep those written deliberately. For example, in a sentence *"Когда типо болеешь и не пошел в универ: "* *("When you are supposedly sick and did not go to university:")* we have word "типо" which is just incorrect form of "типа" and does not carry any emotional or stylistic pallet. The preservation of lexicon of this kind is crucial considering practical value associated with systems trained on such data. In this work, we agreed to let annotators, who are native Russian speakers and passed the language exam, decide whether spelling errors in particular cases need to be corrected given the general instructions (see Section 3.3 for details).

Due to these two observations, we had to manually revise all the candidates that Yandex.Speller suggested.

3.3 Annotation

Next, we set up two-stage annotation project via a crowd-sourcing platform Toloka[11] (Pavlichenko et al., 2021):

1. **Data gathering stage**: we provide the texts with possible mistakes to annotators and ask them to write the sentence correctly;
2. **Validation stage**: we provide annotators with the pair of sentences (source and its corresponding correction from the previous stage) and ask them to check if the correction is right.

The designs of both projects are presented in Figures 4(see Appendix A 7).

We prepared instructions for annotators for each task. The instructions ask annotators to correct misspellings if it does not alter the original style of the text. Instructions do not provide rigorous criteria

[10]https://yandex.ru/dev/speller/
[11]https://toloka.ai/tolokers

Datasets	Sentence	Type
Aranea web-corpus	**Паррикар** говорит: пусть русские приезжают в Индию, веселятся, тратят деньги. ***Parrikar** says: let the Russians come to India, have fun, spend money.*	Proper name
Literature	**Лгание** Муция Сцеволы до сих пор не обличено ***The lies** of Mucius Scaevola have not yet been exposed*	Obsolete word
News	Лидером **антитопа** стал Мэттью Макконахи, звезда «Настоящего детектива». *The leader of the **antitope** was Matthew McConaughey, the star of True Detective.*	Rare word
Social media	Студент отправил файл с **домашкой** и удалил. спрашиваю: где файл? *The student sent a file with **homework** and deleted it. I ask: where is the file?*	Slang
Strategic Documents	**Кмо** - число объектов культурного наследия, по которым проведен мониторинг ***CMO** - number of objects of cultural heritage, for which monitoring was carried out*	Abbreviations

Table 2: False triggered examples of Yandex.Speller across all **Datasets** with **Type** of misleading trigger attached. All sentences are from corresponding datasets. The boldface indicates words that Yandex.Speller considers misspellings.

on the matter of distinguishing the nature of an error in terms of its origin - whether it came from an urge to endow a sentence with particular stylistic features or from unintentional spelling violation since it is time-consuming and laborious to describe every possible case of employing slang, dialect, colloquialisms, etc. instead of proper language. Instructions also do not distinguish errors that come from the geographical or social background of the source. Instead, we rely on annotators' knowledge and understanding of a language since, in this work, the important factor is to preserve the original style of the text.

To ensure we receive qualified expertise, we set up test iteration on a small subset of the data for both stages. We manually validated the test results and selected annotators, who processed at least six samples (2% of the total test iteration) and did not make a single error. After test iteration, we cut 85% and 86% of labellers for gathering and validation stages.

We especially urge annotators to correct mistakes associated with the substitution of the letters "ё" "й" and "щ" for corresponding "е" "и" and "ш" and not to explain abbreviations and correct punctuation errors. Each annotator is also warned about potentially sensitive topics in data (e.g., politics, societal minorities, and religion).

The annotation details are provided in Table 4, and statistics of confidence levels across all datasets on both stages are provided in Table 3.

Datasets	N_{FI}	C_{FI}	N_{FO}	C_{FO}	C_{SI}	N_{SO}	C_{SO}
Aranea web-corpus	985	78.95	859	85.77	96.38	756	97.95
Literature	494	72.56	262	80.32	99.94	260	99.95
News	245	99	245	99	245	99.94	99.95
Social media	281	67.81	208	79.67	99.93	200	99.934
Strategic Documents	284	79.77	86.14	250	99.94	250	99.95

Table 3: Details on the confidence levels on both stages across all datasets. N_{FI} is a number of samples labelled in the first stage. We proceed with samples with confidence above 67% after the first stage and 90% after the second stage. N_{FO} and N_{SO} are the number of texts selected after the first and second stages, respectively. C_{FI}, C_{FO}, C_{SI} and C_{SO} refer to confidence levels calculated on the corresponding stage and subset in %. C_{FI} and C_{FO} are calculated as the expected value of annotators' support of the most popular correction. C_{SI} and C_{SO} are calculated based on aggregation of annotators' skills.[12]

[12]https://toloka.ai/docs/guide/result-aggregation/#aggr-by-skill

Task	IAA	Total	Overlap	N_T	N_{page}	N_C	N_U	ART
Part 1. Test Iteration	77.98	14$	3	7	4	50	96	132
Part 2. Test iteration	89.09	7.9$	3	8	5	46	74	77
Part 1. Gathering	79.10	112$	3	-	4	-	14	165
Part 2. Validation	99.23	92$	3	-	5	-	10	111

Table 4: Details on the data collection projects for the Golden test set. **IAA** refers to the IAA confindence scores, %. IAA of Part 1 is calculated as the expected value of annotators' support of the most popular correction over all labelled texts. IAA of Part 2 is calculated as an average value of confidence scores (see C_{SI} and C_{SO} in Table 3) over all labelled texts. **Total** is the total cost of the annotation project. **Overlap** is the number of votes per example. N_T is the number of training tasks. N_{page} denotes the number of examples per page. N_C is the number of control examples. N_U is the number of users who annotated the tasks. **ART** means the average response time in seconds.

4 Method

To prove the uniqueness and utility of a dataset, we compare distributions of its spelling errors with those of SpellRuEval-2016 (Sorokin et al.,) and synthetically generated misspellings. To generate errors, we employ two approaches. The first is based on the most common spelling errors, statistics and heuristics and can produce corrupted text without any labelled data. The second approach, on the contrary, needs annotated parallel samples to scan source misspellings and try to emulate the spanning errors process to replicate source distributions. We dedicate the following two subsections to describing both methods in more detail.

4.1 Augmentex

For the first time, we present a tool for augmenting text data and conducting black-box attacks on machine-learning models. Augmentex is based on statistical data and heuristics based on human behaviour when using the keyboard and supports two separate augmentation modes:

1. at the character level;
2. at the word level (each of which has 7 and 5 methods, respectively).

You can control the number of augmentations using three primary parameters: min_aug, max_aug and aug_rate. The last is responsible for the number of augmentation applications by specifying the number of percentages of words or characters of the source string to which methods should be applied. The first two arguments set the lower and upper limits of the number of methods applications. These parameters are necessary, as the source string must only remain completely with augmentations or, on the contrary, is not significantly distorted by them. For convenience, batch data processing was done, in which one can specify how many percentages of the source corpus of texts one needs to apply augmentations. So far, the methods have support for the Russian language, but the variety of languages will expand in the future.

Below, in sub-paragraphs 4.1.1 and 4.1.2, we will describe the operation of all methods in detail.

4.1.1 Methods at the character level

The application scenario is the same for all the methods below: based on the parameters described in paragraph 4.1, the integer N is determined. After that, N characters are randomly selected, and one method is applied to each.

Shift method. This method is based on the heuristic that when printing text, a computer user can sometimes press the $shift\ key$ on a keyboard; in this case, a completely different character will be printed. To do this, we have created a dictionary in which the keys are numbers from 0 to 9, and all letters are in uppercase and lowercase. As values for each key, we put the corresponding keys when the $shift\ key$ is pressed. As a result, we got a dictionary power equal to 76: 33 letters in both registers and 10 digits.

Spelling error method. This method is based on statistical error data collected by researchers from the project KartaSlov[13]. The data contains frequent words of the Russian language and variants of their incorrect spelling (both spelling errors and typos). All erroneous spellings are equipped with weights that can estimate the relative frequency of occurrence of specific errors. The obtained error matrix is a matrix of relative frequencies when instead of correctly using the letter X, the letter Y will be mistakenly used. The reason for the error can be either spelling or a typo. There are correct uses along the lines and erroneous ones along the columns. Each row is individually assigned to one by the maximum value. Thus, the most frequent error in each row will weigh 1.0. The heat map can be viewed in Figure 5 in Appendix B 8.

For ease of use, each line was normalized and written into a dictionary, where the key is a letter of the Russian alphabet. The value is a float list of length 33 with the probability of making a mistake in writing a letter from the key. While applying the method to a particular character, we get a list of probabilities and randomly select a new character according to the probabilities in this list.

Typo method. This method is based on heuristics when a computer user misses a key and accidentally touches an adjacent key. We have created a dictionary where the key is 1 of 33 characters of the Russian alphabet or 1 of 10 digits. By default, each character on the keyboard has six neighbours if you do not consider the extreme characters. For example, the character "п" will have 6 neighbors: "е", "н", "р", "и", "м" and "а". Therefore, we put a list containing neighbouring characters as values. When applying the method, adjacent characters are selected equally likely and replaced by the original character.

Method of deleting a character. When calling this method, an empty string is returned instead of the original character.

Random character insertion method. When calling this method, a place for insertion is randomly selected and a random character from the dictionary is inserted (for the Russian language, this is 33 letters).

Character repetition method. The method is based on the heuristic of the key sticking during typing and as a result of the repetition of consecutive characters in the text. It has an additional parameter $mult\_num$, which is responsible for the upper limit of the number of repetitions of the original character. During the application, the number of repetitions is randomly selected from the range of integers from 1 to $mult\_num$, and the original character repeated as many times is returned.

Character permutation method. This method is based on the heuristic that when typing text quickly, the user often confuses the order of pressing the keys, resulting in consecutive characters having the reverse order of writing. We replace the original character with the following places to model human behaviour.

4.1.2 Methods at the word level

The logic of applying the methods will be similar to that described in paragraph 4.1.1, but the word level is used instead of the character. These methods primarily aim to introduce various language errors (lexical errors, agreement, etc.). Some can be used to add spelling errors and typos at the character level (in this paper, we consider only the spelling errors). We present here the description of all the library features, as it's potentially valuable for future research to investigate the imitation of more complex types of errors than orthographic.

Word replacement method. This method is very similar to its character counterpart – The Spelling error method. Only now, the dictionary acts as an error matrix, where the keys are words without errors, and the value is a list of pairs of the form (a word with an error, the probability of writing this word). It has 22187 keys and 4.1 pairs on average. Researchers collected these statistics from KartaSlov, as we mentioned earlier.

Word deletion method. During the application of the method, an empty string is returned instead of the original word. The logic of the work is similar to the method of removing the character.

[13]https://github.com/dkulagin/kartaslov/tree/master/dataset/orfo_and_typos

Word permutation method. This method rearranges two adjacent words in places. It simulates a syntax error when the word order in a sentence is broken.

The method of adding parasite words. A corpus of the most common parasite words from various open sources was collected to implement this method. The cardinality of the set is equal to 70 words. When applied, one of the parasite words is equally likely inserted into a random place in the sentence, which models the illiterate use of words in speech. They clog up the text's meaning, making it indistinct and difficult to understand.

Capital Letter method. This method changes the case of the first letter in the word. It models the incorrect spelling of proper names in the Russian language.

4.2 Statistic-based spelling corruption

The goal of statistic-based corruption is to mimic misspellings distributions scanned from source texts. The algorithm consists of two consecutive parts: analysis of errors in given sentences, which results in corresponding distributions and applying these distributions to correct texts.

This method needs a parallel dataset, where pair of samples consists of a source sentence, which potentially has spelling errors, and a corresponding correct sentence. Datasets for a spellchecking task often come without any annotation on where the error is located in the source sentence. To analyze spelling errors, we have to know their exact positions. It can be achieved either by manual annotation or automatically. In this work, we implement an algorithm that detects the position of misspelling and its category following predefined types of string edits. The idea behind the algorithm is to calculate Levenshtein distances (Levenshtein, 1966) between all the prefixes of the source sentence and correction and traverse it back.

4.2.1 Error analysis

To analyse the errors, we first have to define the notion of spelling error, types of spelling errors and types of distributions that we model. First, in this paper, we accept only one option of proper spelling. All datasets described in the current work are parallel and have corrected sentences for each corresponding sentence with errors. We consider these corrections *proper spelling*. This arrangement is necessary to suggest the following precedents, which result in errors in correct spelling:

- **Insertion**: insertion of a character;
- **Deletion**: deletion of a character;
- **Substitution**: substitution of a character for another non-identical character;
- **Transposition**: switching places of two contiguous characters;
- **Extra separator**: insertion of a gap;
- **Missing separator**: deletion of a gap;

Characters are represented only by letters of the Russian alphabet. We do not include punctuation signs and letters from other languages. We define a spelling error as an event that can be described by one and only one of the listed precedents. We add uniqueness property to the definition of spelling error to avoid interpretations of a particular event as a composition of multiple precedents. For example, the transposition of two contiguous letters gives the same result as substitution of these letters on one another.

Since we defined the notion of spelling error, we can now describe it with corresponding types. We set the type of error as a random variable T, which can take one of the six possible categories. Each category is a precedent. This assumption is correct because we restricted spelling errors to be described by only one of the precedents. Because T takes one of the six possible outcomes, we assume T follows multinoulli distribution D_T. To describe D_T, we have to estimate the probabilities of each outcome. In this paper, we calculate the number of appearances of each error type, normalize them by the total number of misspellings and use these estimates as parametrization for T.

Another important attribute of an error, that should be studied, is its position in a sentence. We calculate the relative position of a misspelling by dividing its absolute position by the number of characters in a sentence. We treat the relative position as a random variable P distributed according to unknown continuous distribution D_P and take values from the interval [0, 1]. For simplicity, we split this interval

into ten equal non-intersecting semi-open subintervals and model the probability that P will fall in one of them. Since the particular value of P can only take one subinterval, we can say that random variable $\hat{P}$, which describes the categorization of P, follows multinoulli distribution $D_{\hat{P}}$. Analogously, we model it by counting encounters of different subintervals and normalizing it to valid discrete distribution. To analyze different types of errors more thoroughly, we consider P and corresponding $\hat{P}$ to be unique for each misspelling type.

The last characteristic we want to keep track of is the number of spelling errors per sentence. The random variable N, which takes integer numbers starting from zero, can describe this characteristic. For simplicity again, we suggest that N follows multinoulli distribution D_N, with the number of possible outcomes equal to the maximum number of errors in a sentence. We use the same procedure to estimate parameters of D_N.

4.2.2 Text corruption

Since we know how to estimate parameters of D_T, $\hat{P}$ for each type of misspelling and D_N, we can use these distributions to corrupt the correct text and expect corresponding distributions of corrupted texts to be similar to those of source texts. We sample the number of misspellings from D_N for each sentence in a corpus of presumably correct sentences. Then for each misspelling, we sample its type from D_T and its subinterval from $\hat{P}$, corresponding to the selected type. To calculate the exact position of an error in a sentence, we scale back the boundaries of subinterval according to the number of characters in a sentence and sample random positions within these boundaries. We check if sampled position satisfies predefined conditions for the particular type of error. For example, we do not allow the deletion of punctuation signs or the insertion of a double gap. If conditions do not hold, we sample position again or skip this misspelling. If position is found, we apply a selected type of error and proceed to the next misspelling or following sentence. The pseudocode for this procedure can be seen in listing 8 in Appendix B 8.

5 Evaluation

The evaluation process is separated into two parts. First, we evaluate our multi-domain dataset and compare misspellings distributions, described in Section 4.2, with corresponding distributions of SpellRuEval-2016 (Sorokin et al.,) to ensure we bring novelty in the field of automatic spelling correction explained by multi-domain nature of the corpus. Second, we want to evaluate methods of spelling corruption proposed in Section 4. These tools aim to mimic human spelling errors to some degree of accuracy. We generate synthetic misspellings with both methods on the correct sentences of our dataset. We then compare synthetic and natural error distributions analogously to the first part of the evaluation.

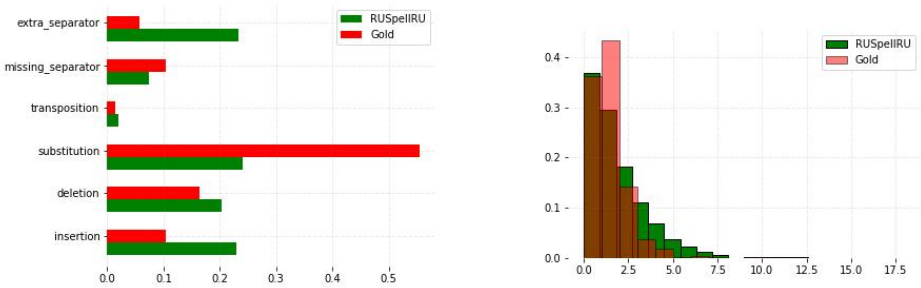
This study primarily focuses on the description and evaluation of proposed methods, rather than conducting a comparative analysis with existing analogues. Specifically, the ButterFingers method is applied to lowercase letters, without considering other symbols or characters. The Case method lacks specific thresholds for incorporating misspellings, resulting in a scenario where the text remains unaltered without any substitutions. Furthermore, it should be noted that the Augmentex tool offers a broader range of perturbation techniques, making it challenging to establish a comprehensive comparison with mentioned tools.

To compare distributions, we employ two approaches: visualization analysis and numeric metrics. The visualization part is represented by histograms that depict distributions of realizations of T, P and N.

We also employ a two-sample variation of Kolmogorov–Smirnov test (Dimitrova et al., 2020) as a numeric metric. Kolmogorov–Smirnov test (Dimitrova et al., 2020) is designed to suit continuous distributions. It does not require normality and can be used with arbitrary distributions and subsets of arbitrary sizes. Thus, in this work, we prefer Kolmogorov–Smirnov test (Dimitrova et al., 2020) over correlation metrics and other tests. It produces scores representing the supremum distance between two empirical distribution functions corresponding to each sample. Then, based on these scores, p-values are calculated under the null hypothesis, which says that two observed sets of values come from the same unknown distribution. We use these p-values in all the tables starting from Table 5 alongside with a significance level of 0.05, which in particular means that if the p-value is less than 0.05, then two given subsets of values do not come from the same underlying distribution.

We apply Kolmogorov–Smirnov test (Dimitrova et al., 2020) for D_P because in Section 4.2, we state that P follows the continuous distribution. N, on the contrary, follows discrete distribution and does not fit in Kolmogorov–Smirnov test (Dimitrova et al., 2020) continuous setup. For N, we use the discrete case of two-sample Kolmogorov–Smirnov test (Dimitrova et al., 2020), and for T, we do not use either of Kolmogorov–Smirnov test (Dimitrova et al., 2020) variations, because some categories are too scarce and estimates may have been incorrect.

5.1 Dataset evaluation



(a) Types of errors

(b) Number of errors per sentence.

Figure 1: The distributions of the errors by type and number in SpellRuEval-2016 and Gold testsets.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SpellRuEval
Aranea	1.0	0.0	0.0	0.0	0.0	0.003	**0**
Literature	0.0	1.0	0.257	0.736	0.0	0.003	**0**
News	0.0	0.239	1.0	0.262	0.0	0.0	**0**
Strategic Documents	0.001	0.743	0.253	1.0	0.0	0.08	**0**
Social media	0.0	0.0	0.0	0.0	1.0	0.0	**0**
Gold	0.001	0.002	0.001	0.08	0.0	1.0	**0**
SpellRuEval	0.0	0.0	0.0	0.0	0.001	0.0	1

Table 5: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for the number of errors per sentence. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **Aranea** refers to Aranea web-corpus, **SpellRuEval** refers to SpellRuEval-2016 (Sorokin et al.,) and **Gold** refers to our dataset. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020).

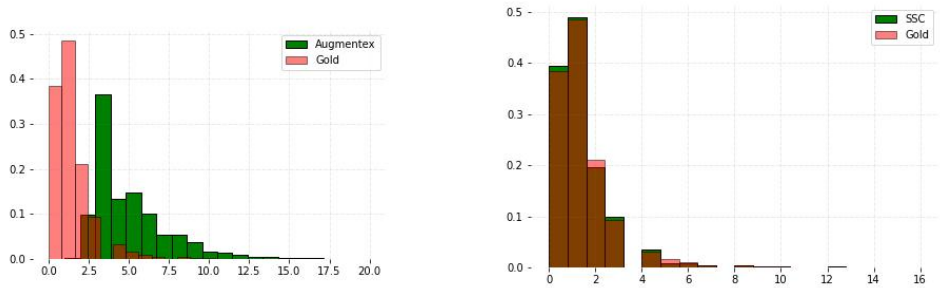
Detailed graphics and tables are in Appendix C 9. Several observations follow an analysis of graphs and tables. First, SpellRuEval-2016 (Sorokin et al.,) and multi-domain dataset seem to deviate in the distribution of types of spelling errors. While the latter has the dominant type of error - *substitution*, - SpellRuEval-2016 (Sorokin et al.,) almost evenly shares misspellings among its four most representative categories. A closer look at the remaining distributions of positions and corresponding tables suggests non-negligible difference in parts of the sentence, where spelling errors occur in the Gold dataset and SpellRuEval-2016 (Sorokin et al.,).

Second, p-values in Tables 5, 6, 8, 11, 12 suggest that Gold dataset differs from its constituents, at least according to corresponding distributions. This observation may be explained by the diverse nature of the source datasets and substantial deviations in properties of errors, which are brought by different domains. This leads to statistics yielded from the Gold dataset, which is a composition of source datasets, to be differ from those gathered from constituents.

Summing up the first part of the experiments, the multi-domain dataset and SpellRuEval-2016 (Sorokin et al.,) are different in proportions of misspellings, their positions in a sentence and domains that are included in corresponding corpora.

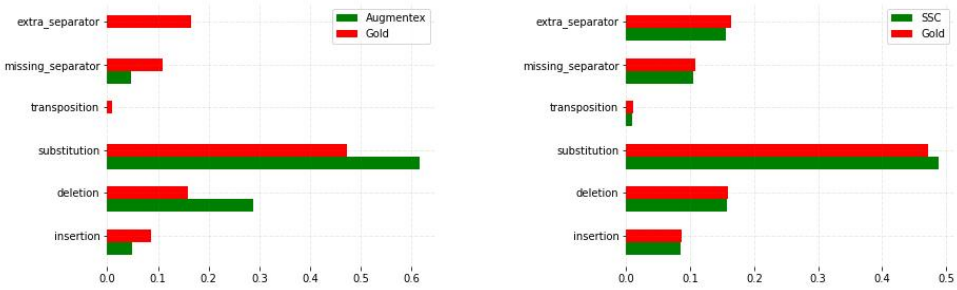
5.2 Spelling corruption methods evaluation

This subsection describes the results of evaluating the proposed spelling corruption methods. We generate synthetic spelling errors with the suggested algorithms on correct sentences of the multi-domain gold dataset. Then we do the same procedure in Section 5.1.



(a) Augmentex (b) SSC

Figure 2: The distributions of number (per sentence) of synthetically generated errors by the proposed methods for spelling corruption compared to the dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



(a) Augmentex (b) SSC

Figure 3: The distributions of types of synthetically generated errors by the proposed methods for spelling corruption compared to the dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.

	Aranea	**Literature**	**News**	**Strategic Documents**	**Social media**	**Gold**	**SSC**	**Augmentex**
Aranea	1.0	0.0	0.0	0.001	0.0	0.002	0.001	0.0
Literature	0.0	1.0	0.227	0.736	0.0	0.001	0.004	0.0
News	0.0	0.231	1.0	0.266	0.0	0.0	0.0	0.0
Strategic Documents	0.0	0.724	0.262	1.0	0.0	0.076	0.12	0.0
Social media	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
Gold	0.001	0.004	0.0	0.079	0.0	1.0	0.85	0.0
SSC	0.001	0.006	0.0	0.122	0.0	0.842	1.0	0.0
Augmentex	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Table 6: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for the number of errors per sentence. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **Aranea** refers to Aranea web-corpus, **SpellRuEval** refers to SpellRuEval-2016 (Sorokin et al.,), **Gold** refers to Gold dataset, **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020).

The detailed graphics and tables are in Appendix D 10. We witness from a visualization point of view that the Statistic-based spelling corruption method fits well for distributions of the gold test set's number and types of spelling errors (see Figure 2 and Figure 3).

However, it should be noticed that we compare two methods on the complete range of sentence lengths. Research on the correlation between sequence length and the number of errors and probable degradation or enhancement of performance of two approaches is yet to be done as a promising aspect of our future work.

Both methods provide mostly high p-values produced by Kolmogorov–Smirnov test (Dimitrova et al., 2020) (see Tables 18, 17, 16, 15, 14, 13) between sets of relative positions of synthetically generated errors and corresponding misspellings from the gold set. Thus, both methods can approximate distributions of human spelling errors.

6 Conclusion

In this paper, we dealt with the spelling errors augmentation problem. We present the multi-domain parallel corpus for the Russian language for the first time. It represents the golden spelling error distribution we compare with the artificial ones. To generate artificial mistakes, we employ two approaches. The first is based on statistics and heuristics and can produce corrupted text without labelled data. The second approach, on the contrary, needs annotated parallel samples to examine source misspellings and replicate the spanning error distributions. The dataset is publicly available in the repository [14]. As part of our future research, we intend to enrich the existing dataset by incorporating data from new domains. Furthermore, an intriguing aspect to explore would be the examination of text distributions pertaining to input sources such as computer keyboards and mobile devices. We propose the inclusion of relevant metadata associated with these sources within the dataset, thereby enhancing its comprehensiveness and contextual relevance.

References

Karel Benes and Lukás Burget. 2020. Text augmentation for language models in high error recognition scenario. *CoRR*, abs/2011.06056.

Vladimír Benko. 2014. Aranea: Yet another family of (comparable) web corpora. *// Text, Speech and Dialogue: 17th International Conference, TSD 2014, Brno, Czech Republic, September 8-12, 2014. Proceedings 17*, P 247–256. Springer.

Eric Brill and Robert C Moore. 2000. An improved error model for noisy channel spelling correction. *// Proceedings of the 38th annual meeting of the association for computational linguistics*, P 286–293.

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Kaustubh D. Dhole, Varun Gangal, Sebastian Gehrmann, Aadesh Gupta, Zhenhao Li, Saad Mahamood, Abinaya Mahendiran, Simon Mille, Ashish Srivastava, Samson Tan, Tongshuang Wu, Jascha Sohl-Dickstein, Jinho D. Choi, Eduard Hovy, Ondrej Dusek, Sebastian Ruder, Sajant Anand, Nagender Aneja, Rabin Banjade, Lisa Barthe, Hanna Behnke, Ian Berlot-Attwell, Connor Boyle, Caroline Brun, Marco Antonio Sobrevilla Cabezudo, Samuel Cahyawijaya, Emile Chapuis, Wanxiang Che, Mukund Choudhary, Christian Clauss, Pierre Colombo, Filip Cornell, Gautier Dagan, Mayukh Das, Tanay Dixit, Thomas Dopierre, Paul-Alexis Dray, Suchitra Dubey, Tatiana Ekeinhor, Marco Di Giovanni, Rishabh Gupta, Rishabh Gupta, Louanes Hamla, Sang Han, Fabrice Harel-Canada, Antoine Honore, Ishan Jindal, Przemyslaw K. Joniak, Denis Kleyko, Venelin Kovatchev, Kalpesh Krishna, Ashutosh Kumar, Stefan Langer, Seungjae Ryan Lee, Corey James Levinson, Hualou Liang, Kaizhao Liang, Zhexiong Liu, Andrey Lukyanenko, Vukosi Marivate, Gerard de Melo, Simon Meoni, Maxime Meyer, Afnan Mir, Nafise Sadat Moosavi, Niklas Muennighoff, Timothy Sum Hon Mun, Kenton Murray, Marcin Namysl, Maria Obedkova, Priti Oli, Nivranshu Pasricha, Jan Pfister, Richard Plant, Vinay Prabhu, Vasile Pais, Libo Qin, Shahab Raji, Pawan Kumar Rajpoot, Vikas Raunak, Roy Rinberg, Nicolas Roberts, Juan Diego Rodriguez, Claude Roux, Vasconcellos P. H. S., Ananya B. Sai, Robin M. Schmidt, Thomas Scialom, Tshephisho Sefara, Saqib N. Shamsi, Xudong Shen, Haoyue Shi, Yiwen Shi, Anna Shvets, Nick Siegel, Damien Sileo, Jamie Simon, Chandan Singh, Roman Sitelew, Priyank Soni, Taylor Sorensen, William Soto, Aman Srivastava, KV Aditya Srivatsa, Tony Sun, Mukund Varma T, A Tabassum, Fiona Anting Tan, Ryan Teehan, Mo Tiwari,

[14] https://huggingface.co/datasets/RussianNLP/russian_multidomain_spellcheck

Marie Tolkiehn, Athena Wang, Zijian Wang, Gloria Wang, Zijie J. Wang, Fuxuan Wei, Bryan Wilie, Genta Indra Winata, Xinyi Wu, Witold Wydmański, Tianbao Xie, Usama Yaseen, M. Yee, Jing Zhang, and Yue Zhang. 2021. Nl-augmenter: A framework for task-sensitive natural language augmentation.

Dimitrina S. Dimitrova, Vladimir K. Kaishev, and Senren Tan. 2020. Computing the kolmogorov-smirnov distribution when the underlying cdf is purely discrete, mixed, or continuous. *Journal of Statistical Software*, 95(10):1–42.

Ilya Gusev. 2020. Dataset for automatic summarization of russian news. // *Artificial Intelligence and Natural Language: 9th Conference, AINL 2020, Helsinki, Finland, October 7–9, 2020, Proceedings 9*, P 122–134. Springer.

Masato Hagiwara and Masato Mita. 2019. Github typo corpus: A large-scale multilingual dataset of misspellings and grammatical errors. *CoRR*, abs/1911.12893.

Vitaly Ivanin, Ekaterina Artemova, Tatiana Batura, Vladimir Ivanov, Veronika Sarkisyan, Elena Tutubalina, and Ivan Smurov. 2020. Rurebus-2020 shared task: Russian relation extraction for business. // *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog" [Komp'iuternaia Lingvistika i Intellektual'nye Tehnologii: Trudy Mezhdunarodnoj Konferentsii "Dialog"]*, Moscow, Russia.

Mark D Kemighan, Kenneth Church, and William A Gale. 1990. A spelling correction program based on a noisy channel model. // *COLING 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. // *Soviet physics doklady*, volume 10, P 707–710. Soviet Union.

V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707, February.

Aouragh Si Lhoussain, Gueddah Hicham, and YOUSFI Abdellah. 2015. Adaptating the levenshtein distance to contextual spelling correction. *International Journal of Computer Science and Applications*, 12(1):127–133.

Nikita Pavlichenko, Ivan Stelmakh, and Dmitry Ustalov. 2021. Crowdspeech and vox diy: Benchmark dataset for crowdsourced audio transcription. // J. Vanschoren and S. Yeung, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

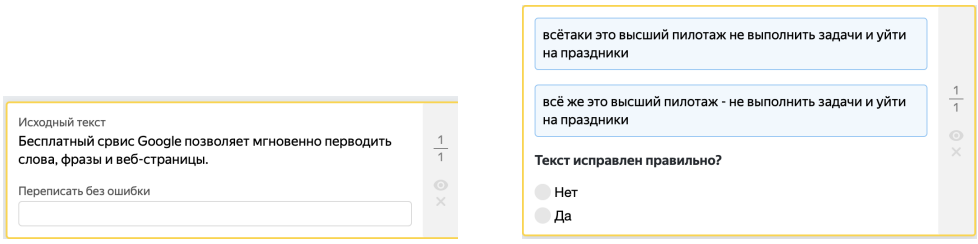
Tatiana Shavrina. 2017. Methods of misspelling detection and correction: A historical overview. *Voprosy Jazykoznanija*, (4):115–134.

AA Sorokin, AV Baytin, IE Galinskaya, and TO Shavrina. Spellrueval: the first competition on automatic spelling correction for russian.

Ekaterina Taktasheva, Tatiana Shavrina, Alena Fenogenova, Denis Shevelev, Nadezhda Katricheva, Maria Tikhonova, Albina Akhmetgareeva, Oleg Zinkevich, Anastasiia Bashmakova, Svetlana Iordanskaia, et al. 2022. Tape: Assessing few-shot russian language understanding. *arXiv preprint arXiv:2210.12813*.

Kristina Toutanova and Robert C Moore. 2002. Pronunciation modeling for improved spelling correction. // *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, P 144–151.

7 Appendix A



(a) Gathering. (b) Validation.

Figure 4: The example of the Yandex.Toloka design settings for the error gathering and validation steps.

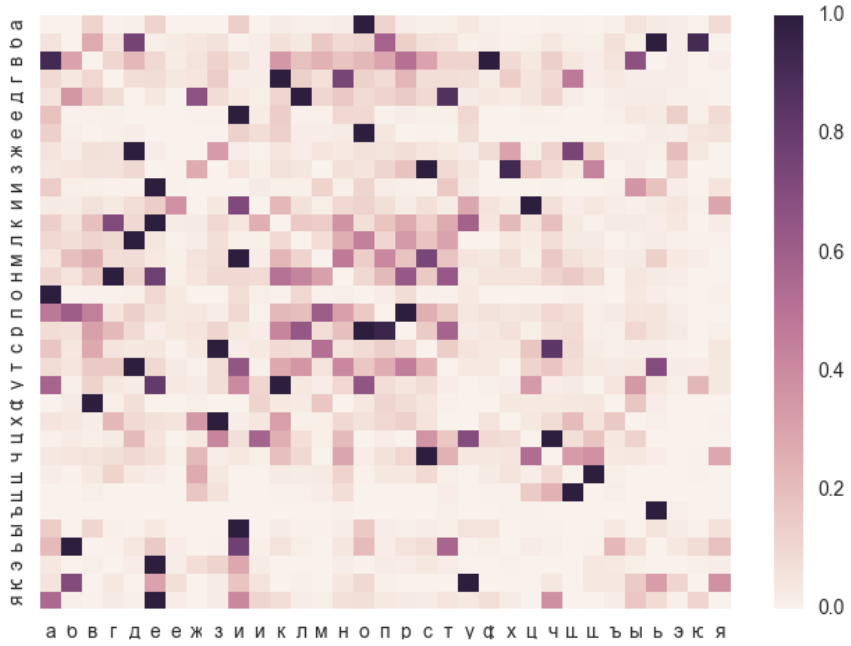


Figure 5: The heat map is read line by line. For example, for the letter "а", the most likely error is "о". All other errors are significantly less likely.

8 Appendix B

```
num_errors = D_N.sample() # sample number of errors
for error in num_errors:
    type = D_T.sample() # sample type of error
    subinterval = D_Ps[type].sample() # sample relative boundaries
    pos_left = len(sentence) * subinterval[0] # rescale boundaries back
    pos_right = len(sentence) * subinterval[1]

    counter = 0
    pos = choice(pos_left, pos_right) # sample position
    while not satisfy(type, pos): # check if conditions hold
        pos = choice(pos_left, pos_right)
        counter += 1
        if counter > max_tries: # if we tried every position in subinterval
            skip = True
            break
    if not skip:
        sentence = apply(sentence, pos, type) # insert the error
    skip = False
```

9 Appendix C

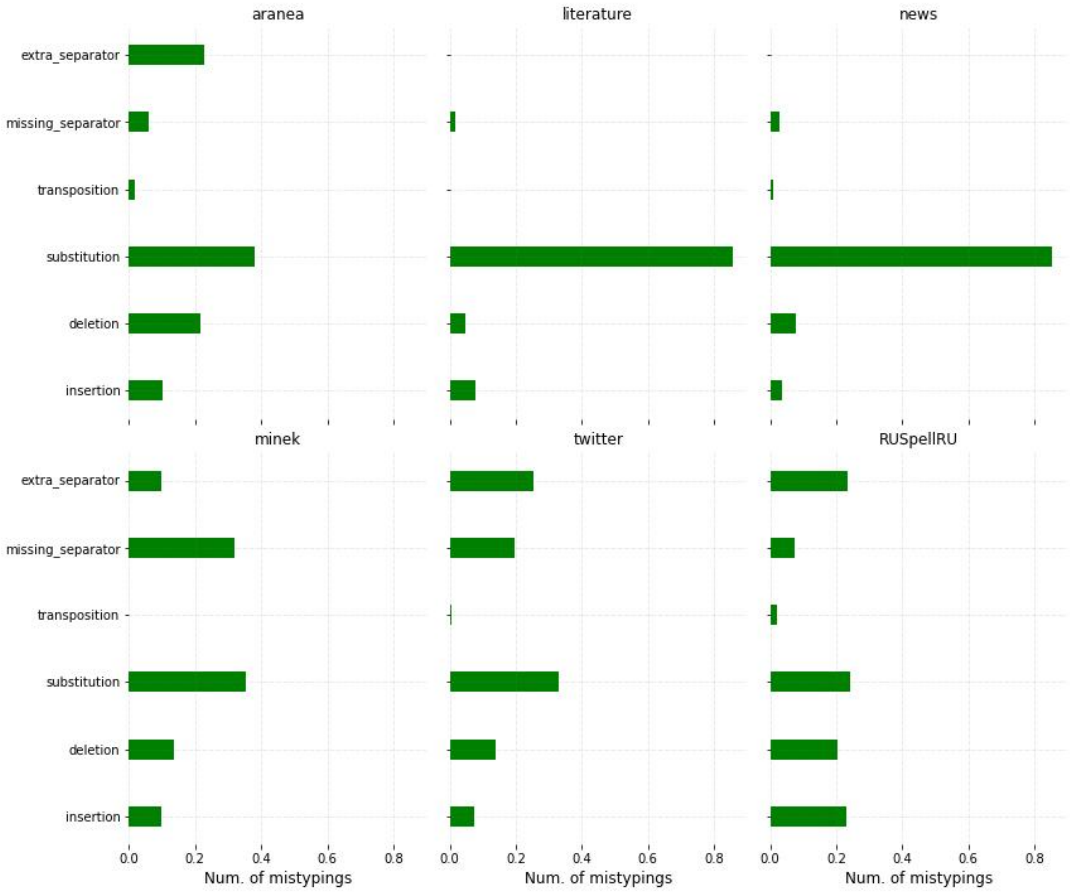


Figure 6: The frequencies of various types of errors encountered in different domains. *aranea, literature, news, minek, twitter* refer to domains of the porposed dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,). It is normalized counters of corresponding error types on the y-axis, which makes them estimates of probabilities of outcomes for T.

	Aranea	**Literature**	**News**	**Strategic Documents**	**Social media**	**Gold**	**SpellRuEval**
Aranea	1.0	0.122	0.536	0.249	0.722	0.983	0.001
Literature	0.122	1.0	0.562	0.389	0.449	0.275	0.522
News	0.536	0.562	1.0	0.842	0.842	0.674	0.227
Strategic Documents	0.249	0.389	0.842	1.0	0.773	0.519	0.009
Social media	0.722	0.449	0.842	0.773	1.0	0.927	0.108
Gold	0.983	0.275	0.674	0.519	0.927	1.0	**0.0**
SpellRuEval	0.001	0.522	0.227	0.009	0.108	**0.0**	1.0

Table 7: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *insertion-type errors*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution.

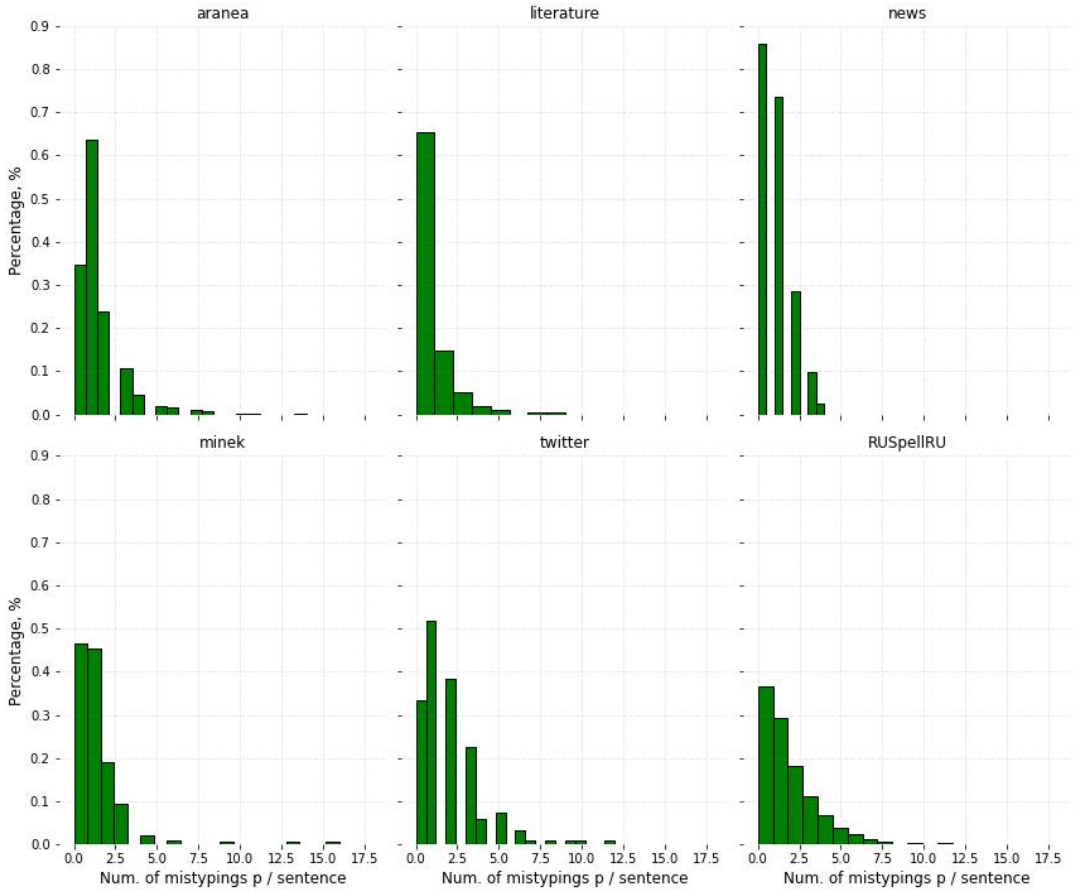
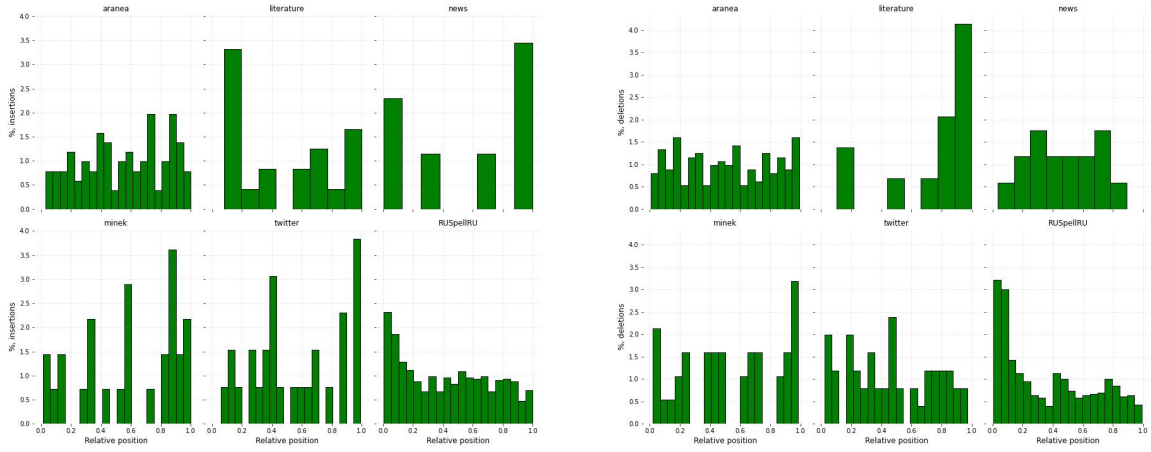


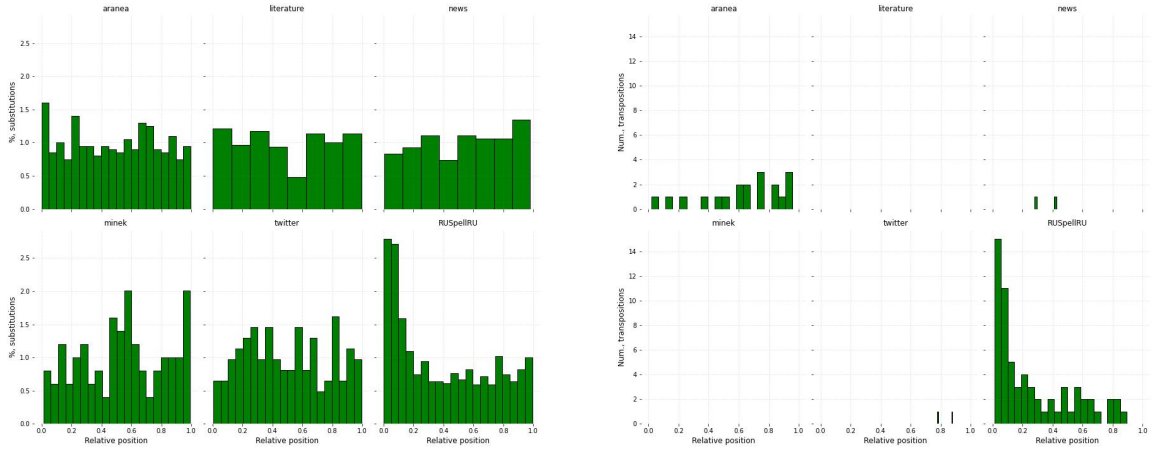
Figure 7: The number of spelling errors across domains in the proposed dataset compared to SpellRuEval-2016 (Sorokin et al.,). *aranea, literature, news, minek, twitter* refer to domains of our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).



(a) Insertions

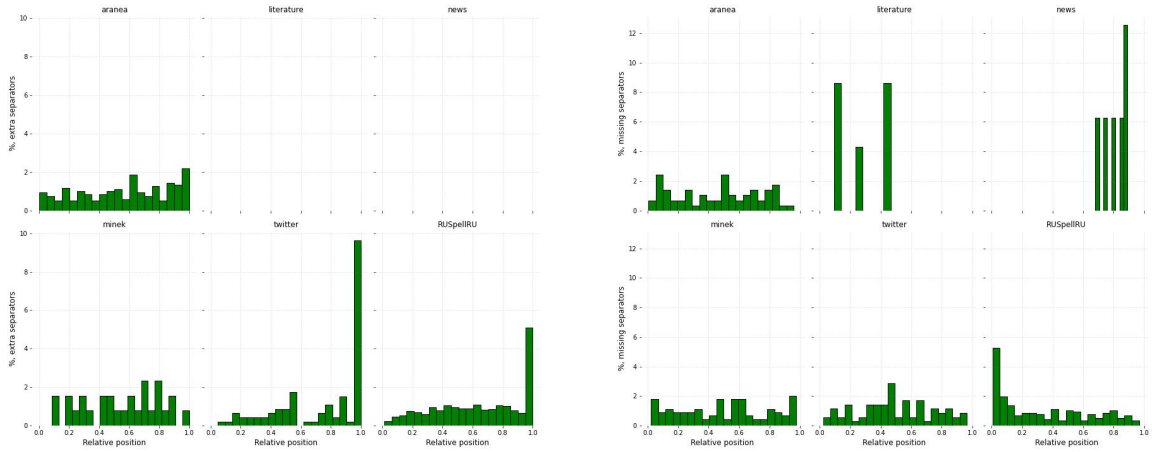
(b) Deletions

Figure 8: Distributions of relative positions of corresponding types of errors across domains in the proposed dataset. *aranea, literature, news, minek, twitter* refer to domains of our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).



(a) Substitutions (b) Transpositions

Figure 9: Distributions of relative positions of corresponding types of errors across domains in the proposed dataset. *aranea, literature, news, minek, twitter* refer to domains of our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).

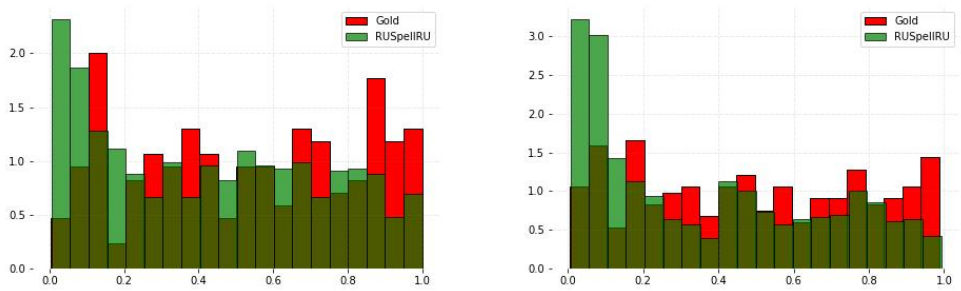


(a) Extra separators (b) Missing separators

Figure 10: Distributions of relative positions of corresponding types of errors across domains in the proposed dataset. *aranea, literature, news, minek, twitter* refer to domains of our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).

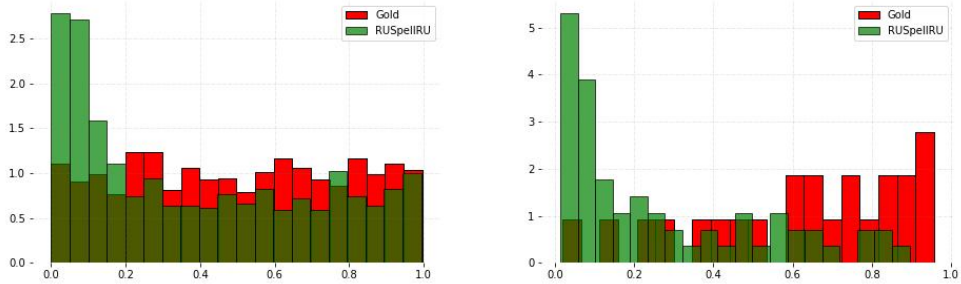
	Aranea	**Literature**	**News**	**Strategic Documents**	**Social media**	**Gold**	**SpellRuEval**
Aranea	1.0	0.002	0.547	0.687	0.479	1.0	**0.0**
Literature	0.002	1.0	0.003	0.015	0.002	0.002	**0.0**
News	0.547	0.003	1.0	0.498	0.838	0.501	0.057
Strategic Documents	0.687	0.015	0.498	1.0	0.318	0.835	0.006
Social media	0.479	0.002	0.838	0.318	1.0	0.48	0.005
Gold	1.0	0.002	0.501	0.835	0.48	1.0	**0.0**
SpellRuEval	**0.0**	**0.0**	0.057	0.006	0.005	**0.0**	1.0

Table 8: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *deletion-type errors*. Table entries are two-tailed p-vals given the null hypothesis that two subsets of samples come from the same distribution.



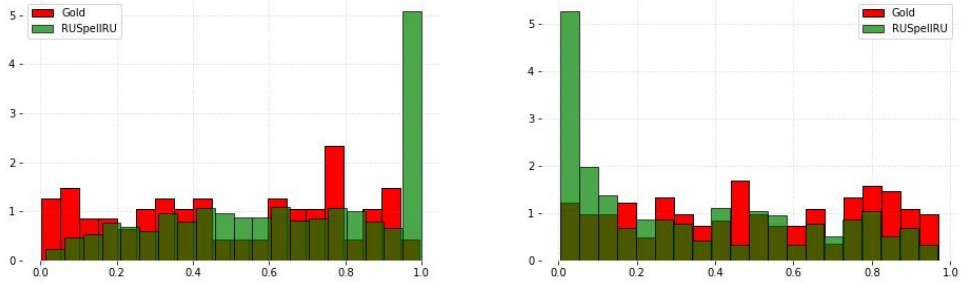
(a) Insertions (b) Deletions

Figure 11: Distributions of relative positions of corresponding types of errors between the multi-domain dataset and SpellRuEval-2016 (Sorokin et al.,). *Gold* refer to our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).



(a) Substitutions (b) Transpositions

Figure 12: Distributions of relative positions of corresponding types of errors between the multi-domain dataset and SpellRuEval-2016 (Sorokin et al.,). *Gold* refer to our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).



(a) Extra separators (b) Missing separators

Figure 13: Distributions of relative positions of corresponding types of errors between the multi-domain dataset and SpellRuEval-2016 (Sorokin et al.,). *Gold* refer to our dataset and *RUSpellRU* refer to SpellRuEval-2016 (Sorokin et al.,).

10 Appendix D

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SpellRuEval
Aranea	1.0	0.911	0.485	0.207	0.421	0.906	**0.0**
Literature	0.911	1.0	0.342	0.086	0.535	0.848	**0.0**
News	0.485	0.342	1.0	0.809	0.592	0.67	**0.0**
Strategic Documents	0.207	0.086	0.809	1.0	0.348	0.241	**0.0**
Social media	0.421	0.535	0.592	0.348	1.0	0.792	**0.0**
Gold	0.906	0.848	0.67	0.241	0.792	1.0	**0.0**
SpellRuEval	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.0

Table 9: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *substitution-type errors*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SpellRuEval
Aranea	1.0	-	0.143	-	0.267	1.0	**0.0**
Literature	-	-	-	-	-	-	-
News	0.143	-	1.0	-	0.333	0.187	0.28
Strategic Documents	-	-	-	-	-	-	-
Social media	0.267	-	0.333	-	1.0	0.3	0.009
Gold	1.0	-	0.187	-	0.3	1.0	**0.0**
SpellRuEval	**0.0**	-	0.28	-	0.009	**0.0**	1.0

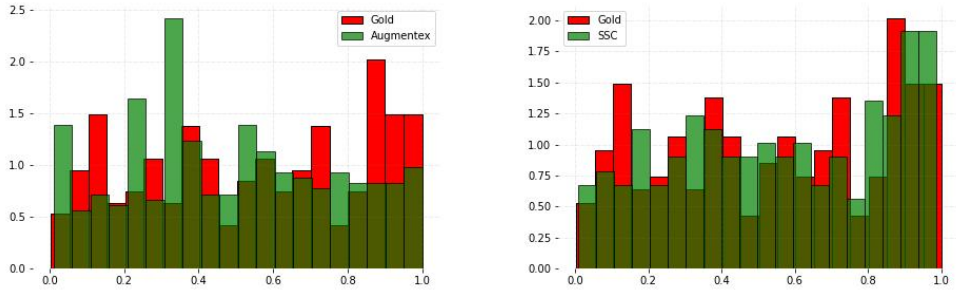
Table 10: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *transposition-type errors*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SpellRuEval
Aranea	1.0	-	-	0.585	0.0	0.066	**0.0**
Literature	-	-	-	-	-	-	-
News	-	-	-	-	-	-	-
Strategic Documents	0.585	-	-	1.0	0.0	0.15	0.046
Social media	0.0	-	-	0.0	1.0	0.0	**0.0**
Gold	0.066	-	-	0.15	0.0	1.0	0.003
SpellRuEval	**0.0**	-	-	0.046	**0.0**	0.003	1.0

Table 11: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *extra separators*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. Gaps indicate that samples from this domain are absent.

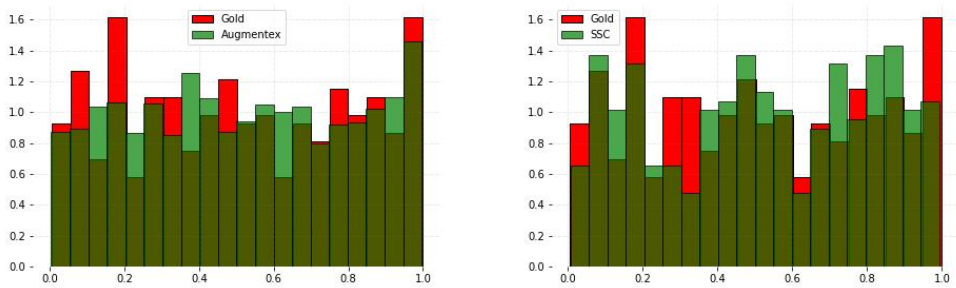
	Aranea	Literature	News	Strategic Documents	Social media	Gold	SpellRuEval
Aranea	1.0	0.071	0.002	0.63	0.459	0.917	0.008
Literature	0.071	1.0	0.004	0.056	0.074	0.057	0.502
News	0.002	0.004	1.0	0.002	0.001	0.001	**0.0**
Strategic Documents	0.63	0.056	0.002	1.0	0.658	0.983	**0.0**
Social media	0.459	0.074	0.001	0.658	1.0	0.808	**0.0**
Gold	0.917	0.057	0.001	0.983	0.808	1.0	**0.0**
SpellRuEval	0.008	0.502	**0.0**	**0.0**	**0.0**	**0.0**	1.0

Table 12: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *missing separators*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. Gaps indicate that samples from this domain are absent.



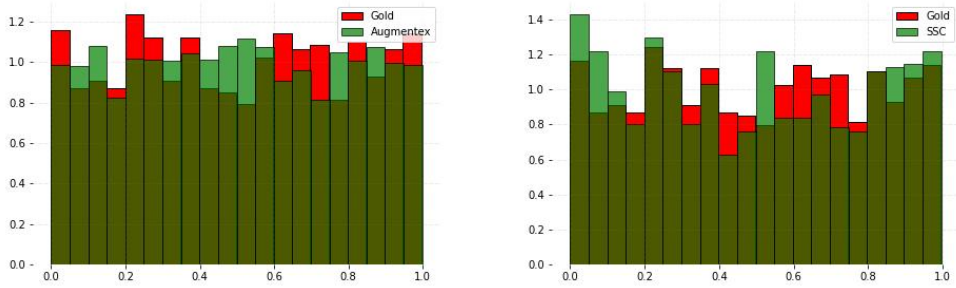
(a) Augmentex (b) SSC

Figure 14: Distributions of relative positions of synthetically generated insertions by the proposed methods for spelling corruption compared to *insertions* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



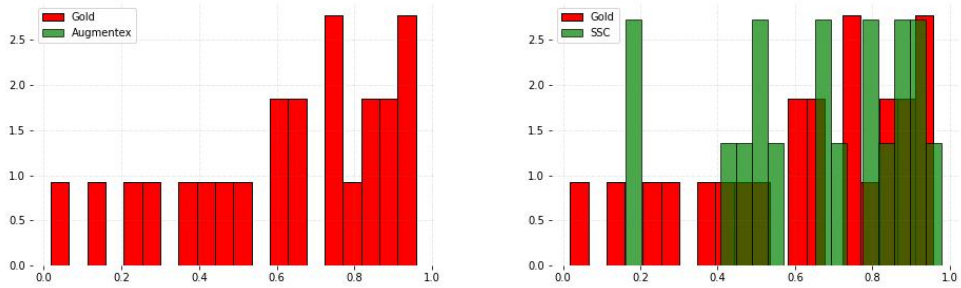
(a) Augmentex (b) SSC

Figure 15: Distributions of relative positions of synthetically generated deletions by the proposed methods for spelling corruption compared to *deletions* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



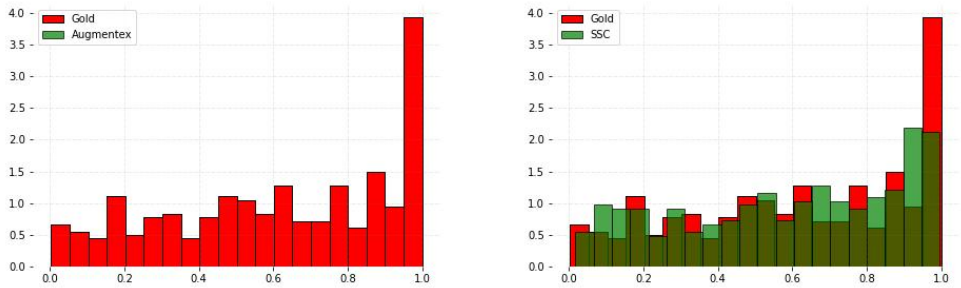
(a) Augmentex (b) SSC

Figure 16: Distributions of relative positions of synthetically generated substitutions by the proposed methods for spelling corruption compared to *substitutions* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



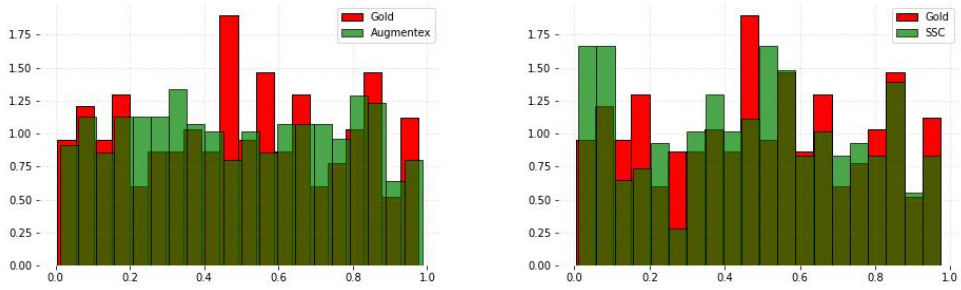
(a) Augmentex (b) SSC

Figure 17: Distributions of relative positions of synthetically generated transposition by both of the proposed methods for spelling corruption compared to *transposition* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



(a) Augmentex (b) SSC

Figure 18: Distributions of relative positions of synthetically generated extra separators by the proposed methods for spelling corruption compared to *extra separators* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.



(a) Augmentex (b) SSC

Figure 19: Distributions of relative positions of synthetically generated missing separators by the proposed methods for spelling corruption compared to *missing separators* in dataset. *Augmentex* and *SSC* refer to the methods described in Section 4.1 and Section 4.2 respectively and *Gold* refers to multi-domain dataset.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	0.122	0.536	0.249	0.722	0.983	0.738	0.077
Literature	0.122	1.0	0.562	0.389	0.449	0.275	0.146	0.16
News	0.536	0.562	1.0	0.842	0.842	0.674	0.801	0.316
Strategic Documents	0.249	0.389	0.842	1.0	0.773	0.519	0.479	0.023
Social media	0.722	0.449	0.842	0.773	1.0	0.927	0.903	0.51
Gold	0.983	0.275	0.674	0.519	0.927	1.0	0.924	0.017
SSC	0.738	0.146	0.801	0.479	0.903	0.924	1.0	0.021
Augmentex	0.077	0.16	0.316	0.023	0.51	0.017	0.021	1.0

Table 13: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *insertions*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020).

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	0.002	0.547	0.687	0.479	1.0	0.49	0.79
Literature	0.002	1.0	0.003	0.015	0.002	0.002	0.003	0.001
News	0.547	0.003	1.0	0.498	0.838	0.501	0.41	0.569
Strategic Documents	0.687	0.015	0.498	1.0	0.318	0.835	0.686	0.789
Social media	0.479	0.002	0.838	0.318	1.0	0.48	0.204	0.294
Gold	1.0	0.002	0.501	0.835	0.48	1.0	0.574	0.796
SSC	0.49	0.003	0.41	0.686	0.204	0.574	1.0	0.51
Augmentex	0.79	0.001	0.569	0.789	0.294	0.796	0.51	1.0

Table 14: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *deletions*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020).

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	0.911	0.485	0.207	0.421	0.906	0.406	0.562
Literature	0.911	1.0	0.342	0.086	0.535	0.848	0.742	0.583
News	0.485	0.342	1.0	0.809	0.592	0.67	0.233	0.179
Strategic Documents	0.207	0.086	0.809	1.0	0.348	0.241	0.135	0.231
Social media	0.421	0.535	0.592	0.348	1.0	0.792	0.273	0.72
Gold	0.906	0.848	0.67	0.241	0.792	1.0	0.139	1.0
SSC	0.406	0.742	0.233	0.135	0.273	0.139	1.0	1.0
Augmentex	0.562	0.583	0.179	0.231	0.72	1.0	1.0	1.0

Table 15: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *substitutions*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020).

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	-	0.143	-	0.267	1.0	0.976	-
Literature	-	-	-	-	-	-	-	-
News	0.143	-	1.0	-	0.333	0.187	0.063	-
Strategic Documents	-	-	-	-	-	-	-	-
Social media	0.267	-	0.333	-	1.0	0.3	0.474	-
Gold	1.0	-	0.187	-	0.3	1.0	0.941	-
SSC	0.976	-	0.063	-	0.474	0.941	1.0	-
Augmentex	-	-	-	-	-	-	-	-

Table 16: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *transpositions*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020). Gaps indicate that samples from this domain are absent.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	-	-	0.585	0.0	0.066	0.572	-
Literature	-	-	-	-	-	-	-	-
News	-	-	-	-	-	-	-	-
Strategic Documents	0.585	-	-	1.0	0.0	0.15	0.259	-
Social media	0.0	-	-	0.0	1.0	0.0	0.0	-
Gold	0.066	-	-	0.15	0.0	1.0	0.0	-
SSC	0.572	-	-	0.259	0.0	0.0	1.0	-
Augmentex	-	-	-	-	-	-	-	-

Table 17: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of *extra separators*. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020). Gaps indicate that samples from this domain are absent.

	Aranea	Literature	News	Strategic Documents	Social media	Gold	SSC	Augmentex
Aranea	1.0	0.071	0.002	0.63	0.459	0.917	0.976	0.833
Literature	0.071	1.0	0.004	0.056	0.074	0.057	0.093	0.092
News	0.002	0.004	1.0	0.002	0.001	0.001	0.001	0.003
Strategic Documents	0.63	0.056	0.002	1.0	0.658	0.983	0.707	0.87
Social media	0.459	0.074	0.001	0.658	1.0	0.808	0.454	0.559
Gold	0.917	0.057	0.001	0.983	0.808	1.0	0.477	0.701
SSC	0.976	0.093	0.001	0.707	0.454	0.477	1.0	0.298
Augmentex	0.833	0.092	0.003	0.87	0.559	0.701	0.298	1.0

Table 18: Kolmogorov–Smirnov test (Dimitrova et al., 2020) p-values for relative positions of missing separators. Table entries are two-tailed p-values given the null hypothesis that two subsets of samples come from the same distribution. **SSC** and **Augmentex** are methods described in Section 4.1 and Section 4.2 respectively. Reported values are averaged over 5 runs of the Kolmogorov–Smirnov test (Dimitrova et al., 2020). Gaps indicate that samples from this domain are absent.