Paraphrase Detection Using Machine Translation and Textual Similarity Algorithms

Dmitry Kravchenko(✉)

Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva, Israel
to.dmitry.kravchenko@gmail.com

Abstract. I present experiments on the task of paraphrase detection for Russian text using Machine Translation (MT) into English and applying existing sentence similarity algorithms in English on the translated sentences. But since I use translation engines - my method to detect paraphrases can be applied to any other languages, which translation into English is available on translation engines. Specifically, I consider two tasks: given pair of sentences in Russian – classify them into two (non-paraphrases, paraphrases) or three (non-paraphrases, near-paraphrases, precise-paraphrases) classes. I compare five different well-established sentence similarity methods developed in English and three different Machine Translation engines (Google, Microsoft and Yandex). I perform detailed ablation tests to identify the contribution of each component of the five methods, and identify the best combination of Machine Translation and sentence similarity method, including ensembles, on the Russian Paraphrase data set. My best results on the Russian data set are an Accuracy of 81.4% and F1 score of 78.5% for an ensemble method with the translation using three MT engines (Google, Microsoft and Yandex). This compares favorably with state of the art methods in English on data sets of a similar size which are in the range of Accuracy 80.41% and F1-score of 85.96%. This demonstrates that, with the current level of performance of public MT engines, the simple approach of translating/classifying in English has become a feasible strategy to address the task. I perform detailed error analysis to indicate potential for further improvements.

Keywords: Paraphrase detection · Semantic similarity algorithms
Machine translation · Supervised classification

1 Introduction

1.1 Motivation

Paraphrase identification is useful in many natural language applications such as search engines (to calculate relevance of one sentence to the other), in plagiarism detection systems, authorship identification, patents and copyright detection systems, question-answering bots (to compute the semantic similarity between a sentence given by a human and sentences stored in a corpus database), and text summarization. This task consists of determining whether two sentences convey similar content to the extent that one can held as a re-statement of the other.

The task has been well studied in English, and methods have been developed that reach pretty good results. These methods are not easily applicable directly to other languages, because they rely on rich lexical resources such as thesauri and large-scale word embedding which are not available in many other languages.

In this paper, I report on experiments to assess the feasibility of a simple strategy to adapt existing techniques in English to a Russian data set of paraphrase sentences: I first translate the Russian sentences into English sentences using publicly available Machine Translation (MT) engines (I test Google Translate, Microsoft Bing and Yandex) and then apply a variety of English techniques on the translated sentences to establish their paraphrase relation.

I find that this simple strategy provides "good enough" results on the data set for little effort, especially when compared with the complexity of acquiring large coverage thesauri in Russian and/or training statistical models on large amounts of Russian text.

1.2 Objective

My objective is to establish the feasibility of applying the strategy of MT as a preprocessing step to address the paraphrase detection task. Clearly, using a translation engine introduces noise because existing MT engines have limited accuracy. I compare three different MT engines to control for this aspect.

1.3 Task Description

Given the Russian paraphrases data set – the goal is to compute sentence similarity. The task is cast as two distinct classification tasks: (1) separate the list of sentence pairs into three classes: non-paraphrases, near-paraphrases and precise-paraphrases; (2) classify the pairs into two classes: non-paraphrases and paraphrases. Results are measured by two scores: F1 score and Accuracy. I use the shared task data set distributed at the Workshop of the International conference in Artificial Intelligence and Natural Language - AINL 2016.

2 Related Work

For English - there are two main paraphrase data sets: the Microsoft Research Paraphrase Corpus (MSRP) [12] and PPDB: the Paraphrase Database [11]. MSRP contains 5801 pairs of sentences (4076 are in the training set, and 1725 are in the test set), which are classified by humans into two classes: paraphrases and non-paraphrases. The highest achievement on MSRP is recorded by Ji and Eisenstein (2013), who used a method of matrix factorization with supervised reweighting, and which achieved an Accuracy of 80.41% and an F1 score of 85.96%. Another notable result was achieved by Socher et al. (2011), which achieved an Accuracy of 76.8% and an F1 score of 83.6%. Recent application of deep learning to the task is reported in Yin and Schutze (2015), where they applied convolutional neural networks and achieved an Accuracy of 78.4% and an F1 score of 84.6%.

For Russian corpus in one of the recent publications [10] Pronoza et al. (2015) achieved F1 score of 82.46% on binary classification task in paraphrase detection.

Madnani et al. (2012) [19] in their paper *Re-examining Machine Translation Metrics for Paraphrase Identification*, for solving the task of this paper used only Machine Translation metrics like BLEU(1-4), MAXSIM, BADGER, SEPIA, TER, NIST(1-5), METEOR, TERp, and has achieved on MSRP corpus 77.4% in Accuracy and 84.1% in F1 score, proving by this the effectiveness of these metrics. In my paper I will use only BLEU scores, and all rest would be semantic similarity algorithms scores. And machine translation I would use to prepare input data for these similarity algorithms.

Paraphrase detection is closely related to the task of textual entailment (TE) identification [17]. TE is a directed relationship between text and hypothesis. Bidirectional TE have not reached the same level of performance as direct paraphrase detection in English.

3 Data Set

Input data is a list of pairs of sentences in Russian which are collected from news headlines. The training set includes 7,227 pairs of sentences, which are classified by humans into three classes: 2,582 non-paraphrases, 2,957 near-paraphrases, and 1,688 precise-paraphrases.

Experimental settings: 14,454 sentences with approximately 117,000 words, in which approximately 23,000 words have unique forms. Sentence length is 8 words on average.

Output data is the list of predicted classes for each one of two tasks (described above), which is assigned to each pair of the sentences of the test part of cross-validation test.

4 Baseline

4.1 Algorithm

As a baseline algorithm I use the standard BLEU sentence similarity metric which can get input in Russian, and doesn't require translation into English.

BLEU scores with smoothing methods are from [2] with word n-grams. It is mentioned in [2] that original BLUE scores required no smoothing, as they were developed for document-level classification. But for the sentence-level classification they used these smoothing techniques.

These two smoothing techniques work as follows: assume that I match word n-grams for $n = 1...N$ (usually, $N = 4$). Let m_n be the count of the matching words in both sentences, and let $\widehat{m_n}$ be the modified n-gram match count.

Smoothing 1 is defined as follows: if there are no matched words in n-grams, then I use a small positive value ϵ to replace the 0 for n $\in$ [1..N]. if $m_n = 0$ then $\widehat{m_n} = \epsilon$.

Smoothing 2 (proposed by Lin and Och, 2004) is defined as follows: I add 1 to the matched n-gram count and the total n-gram count for n ranging from 2 to N.

Formally: for $n \in [2..N]$ I calculate: $\widehat{m_n} = m_n + 1$, and $\widehat{l_n} = l_n + 1$

4.2 Results

Table 1 contains execution results of BLEU algorithms on First Task (3-way classification) and Second task (2-way classification) with word n-grams.

Table 1. BLEU of two smoothing types

BLEU smoothing type	First task		Second task	
	Accuracy	F1 score	Accuracy	F1 score
Type 1 (1-g)	**57.43**	**55.07**	**76.64**	**71.33**
Type 1 (2-g)	55.76	52.83	73.72	70.76
Type 1 (3-g)	55.27	51.54	73.66	70.10
Type 1 (4-g)	49.50	45.50	64.28	41.70
Type 2 (1-g)	57.43	55.07	76.64	71.33
Type 2 (2-g)	56.81	53.44	76.54	70.57
Type 2 (3-g)	56.45	52.71	76.26	70.57
Type 2 (4-g)	56.33	52.20	76.17	70.67

5 Algorithm

5.1 Brief Explanation

In Appendix A, you can see the figure of algorithm data-flow.

Many sentences in the data set (which are news feeds) contain acronyms. I substitute acronyms to their full names using acronyms list derived from www.wiktionary.org. Acronym expansion is performed on the Russian sentences before they are translated.

I then translate the sentences with expanded acronyms from Russian to English. I used translation engines APIs to do so.

Finally, I construct a feature vector for the classifier - using a variety of sentence similarity algorithms which compute similarity scores on pairs of sentences.

These feature vectors I use as an input to GradientBoosting classifier and after computation it gives the class prediction for each pair of sentences.

5.2 Detailed Description

Step 1: Preprocessing I substitute all of the acronyms in the sentences to their full names. This step is very important because all of the toolkits which I use do not recognize Russian acronyms, particularly after they are translated to English. Expanding acronyms helps the MT engine and the sentence similarity methods process all words with better access to the meaning as opposed to the acronym. I used as an acronyms dictionary - online thesaurus https://www.wiktionary.org/. On the training set it had coverage of 47% of the acronyms.

Step 2: After substituting acronyms, I translate Russian to English. I used 3 online translation engines: Google, Microsoft and Yandex. Each of these MT engines has it's own APIs to receive an original sentences, and and send back a translated ones. Each of the MT engines gave its own translation, most of the time slightly different from one another. Each of the translations gave different score results when passed to the sentence similarity toolkits. On the given corpus of sentences, Yandex and Google showed the most accurate translation as measured by the classification performance downstream (higher F1 score and Accuracy scores).

Step 3: Running sentence similarity toolkits on the pairs of sentences, translated to English, and getting scores on each pair, saving them into a json data set file. I use six distinct semantic similarity toolkits for first task, and five - for second task, which are described below.

Step 4: Train a classifier: I train a Gradient Booster classifier algorithm, fed with the vectors of sentence similarity measures. This classifier is comparable to Support Vector Machines in its method, and it gives better results both in F1 measure and Accuracy on our corpus. This method consists of learning an ensemble classifier which combines the similarity scores of five English sentence similarity methods.

I use the Scikit-learn implementation of Gradient Boosting. The following Python code shows the specific parameters I used:

```
import sklearn.ensemble
clf=sklearn.ensemble.GradientBoostingClassifier(n_estimators=100,
max_depth=3)
```

I chose Gradient Boosting classifier, since it gave more accurate classification than either SVM with Gaussian Kernel, or Random Forest. In both cases it gave more then 1 percent to F1 score and Accuracy, then two mentioned classifiers.

My feature vector includes 77 features (which would be described in detail further) for the First Task (3-way classification: 77 features = 23 features * 3 translate engines + 8 BLEU features) and 69 features (described in detail further) for the Second Task (2-way classification: 69 features = 23 features * 3 translations).

The difference between the number of features used in the two tasks is because I did not include BLEU scores to solve the Second Task since these scores worsened class recognition and led to lower F1 score and Accuracy. BLEU scores did help on the First Task, and improved classification.

5.3 Feature Vector Structure for Each One of the Three Translations

Toolkits:

1. SEMILAR [4–7]
   - Number of used features: 6
   - Feature names: bleuComparer, cmComparer, dependencyComparerWnLeskTanim, greedyComparerWNLin, lsaComparer, optimumComparerLSATasa
2. DKPro Similarity [9]
   - Number of used features: 13
   - Feature names: CosineSimilarity, ExactStringMatchComparator, GreedyStringTiling 2-g, GreedyStringTiling 4-g, JaroSecondStringComparator, JaroWinklerSecondStringComparator, normalized LevenshteinComparator, LongestCommonSubsequenceNormComparator, SubstringMatchComparator, WordNGramContainmentMeasure, WordNGram JaccardMeasure 2-g, WordNGramJaccardMeasure 3-g, WordNGramJaccardMeasure 4-g
3. Python difflib
   - Number of used features: 1
   - Feature name: difflib SequenceMatcher comparator

   Example of code using it:
   ```
   import difflib
   sm = difflib.SequenceMatcher(None)
   sm.set_seq1('sentence one')
   sm.set_seq2('sentence two')
   print sm.ratio()
   ```
4. Algorithm of [1]
   - Number of used features: 2
   - Feature names: Sentence similarity scores
5. Swoogle [3]
   - Number of used features: 1
   - Swoogle comparator
6. BLEU scores (on the Russian version of the sentences, no need for English translation) [2]
   - Number of used features: 8
   - Feature names: BLEU with smoothing method number 1 (described in [2]) (1/2/3/4-g), BLEU with smoothing method number 2 (described in [2]) (1/2/3/4-g)

6 Comparison of Toolkits on First Task (3-Way Classification)

6.1 Results

To understand which of the scores separately gave more recognition rate - I did 5-fold cross-validation for First Task (3 class classification) on the training set, on all three translations (Google, Microsoft, Yandex) and the results are in Table 2.

Table 2. Toolkits results

Toolkit	Accuracy	F1 score
SEMILAR	**62.26**	**60.15**
DKPro Similarity	61.14	59.30
Python difflib	57.07	53.51
Algorithm of [1]	60.02	57.66
Swoogle	59.15	55.52
BLEU	57.34	54.96
All six toolkits together	**64.14**	**62.46**

6.2 Confusion Matrix

In Table 3 is a confusion matrix, which I got by running First Task using all six toolkits together:

Table 3. Confusion matrix

	Non-paraphrases	Near-paraphrases	Precise-paraphrases
Non-paraphrases	1751	798	33
Near-paraphrases	444	2164	349
Precise-paraphrases	74	893	721

As it can be seen from the matrix that two major mistakes in classification are:

– of non-paraphrases, which were incorrectly classified as near-paraphrases (798 pairs), which is 30.9% of total amount. Cases for such an errors are described in 'error analysis' section;

– of precise-paraphrases, which were incorrectly classified as near-paraphrases (893 pairs). Also can be noticed that only 721 precise-paraphrases were classified correctly, which is only 42.71% of total amount. This shows us that for semantic similarity algorithm it is hard to distinguish between near-paraphrases and precise-paraphrases, which could be explained that in fact precise-paraphrases have just a light semantic difference from near-paraphrases.

7 Ablation Test and Its Analysis on Second Task (2-Way Classification)

7.1 Results

To understand which of the scores gave more effect to the result - I did 5-fold cross-validation for Second Task (2 class classification) on the training set, on all three translations (Google, Microsoft, Yandex).

For each on the following experiments I combined feature vectors for the classifier as a concatenation of feature vectors of relevant toolkits.

Since it is the Second task I did not include BLEU scores (because they are not improving the results).

The following result are random combinations of toolkits, chosen in such a way - to cover most of the cases Tables 10, 11, 12 and 13 in Appendix B.

It can be seen from the Table 10 (appendix) that all of the toolkits give Accuracy between 75.92% and 80.13% and F1 score between 71.36% and 77.02%. By combining scores (Tables 11, 12 and 13) of these toolkits together I achieve maximum of **81.41%** in Accuracy and **78.51%** in F1 score. If I take as the basis scores from SEMILAR toolkit - by adding other scores to feature vector I achieve improvement of 1.28% in Accuracy and 1.49% in F1 score. Note that each time by adding more scores from one more toolkit to the feature vector - I improve the result, hence I chose optimal scores.

7.2 Confusion Matrix

In Table 4 is a confusion matrix, which I got by running Second Task using all five toolkits together:

Table 4. Confusion matrix

	Non-paraphrases	Paraphrases
Non-paraphrases	1612	970
Paraphrases	373	4272

The matrix indicates that the major source of error is the classification of non-paraphrases, which were mistakenly classified as paraphrases (970 pairs), which is 37.57%. The sources for such an errors are described in ‘error analysis’ section.

7.3 Identifying Best SEMILAR Toolkit Score

To detect the best SEMILAR toolkit score, I ran each of them separately, with the following results (Table 5):

Table 5. Score from SEMILAR toolkit

Score name	Accuracy	F1 score
bleuComparer	66.30	65.50
cmComparer	78.42	74.53
dependencyComparerWnLeskTanim	75.77	70.57
greedyComparerWNLin	**79.18**	**75.74**
lsaComparer	70.11	64.82
optimumComparerLSATasa	78.87	75.06

It can be seen that the highest results both in Accuracy and F1 score gave greedyComparerWNLin score. Let's recall that all 13 scores from DKPro Similarity toolkit gave us approximately the same result: Accuracy of 79.52 and F1 score of 75.78. So on the given corpus this one SEMILAR score alone gives approximately the same recognition rate as 13 scores from DKPro, which is impressive.

GreedyComparerWNLin score refers to a sentence to sentence similarity method which greedily aligns words between two sentences. The word alignment method used is WordNet based method proposed by Lin (1998) [16]. The method is described in [4].

8 Comparison of Translation Engines for Second Task (2-Way Classification)

In this section, I compare all three translation engines (Google, Microsoft and Yandex), and conclude which of them gave better F1 score and Accuracy (Table 6):

Table 6. Translation engines comparison

Toolkits	Google		Microsoft		Yandex	
	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score
SEMILAR	78.95	75.19	78.41	74.71	78.40	74.67
DKPro Similarity	78.38	74.30	77.84	73.83	78.74	74.79
Python difflib	74.88	70.04	74.05	68.70	75.05	71.40
Algorithm of [1]	76.72	72.15	76.32	72.39	77.05	72.82
Swoogle	77.94	73.12	77.95	72.80	77.69	73.02
All five toolkits together	**79.90**	**76.52**	**79.25**	**75.76**	**79.93**	**76.53**

Google and Yandex give similar results in translating from Russian to English on our corpus, both better than Microsoft's MT.

9 Error Analysis

Let's examine common mistakes our scores (algorithms) make in giving higher/lower values, causing the classifier to mistakenly predict the wrong class. Provided below sentences in Russian are from AINL 2016 paraphrase shared task corpus (available on http://www.paraphraser.ru/download/).

9.1 False Positive: Mistakenly Predicted as 'Paraphrase'

Such pairs of sentences typically contain for the most part *the same words* (or similar in the meaning), except for a few words which make all the difference, changing the meaning of the sentence completely.

The following tables are different cases of such a words (Tables 7 and 8):

Table 7. Different words are antonyms or different in the meaning words

Sentence pairs (with corresponding English translations) and explanations
Gogol center will show a video of the controversial performance of Tannhauser./ "Гоголь-центр"покажет видеозапись скандального спектакля "Тангейзер"
Kekhman banned Gogol-center to show the video recording of Tannhauser./ Кехман запретил «Гоголь-центру» показывать видеозапись «Тангейзера»
in first sentence: will show video in second sentence: banned to show video
The leader of the British labour party retained the seat in Parliament./ Лидер британских лейбористов сохранил место в парламенте.
The leader of the British labour party resigned because of the defeat in the elections./ Лидер британских лейбористов подал в отставку из-за поражения на выборах.
in first sentence: retained the seat in second sentence: resigned

Idea How to Solve Such Cases

(1) *Difference feature:* Different words are antonyms or different in the meaning words.
Idea for solution: Create a score (algorithm) which checks to which objects (or concepts) are related those antonyms (or different in meaning words), and if they are related to the same objects (or concepts) - this is a sign that different meaning in the pair of sentences exists, so the algorithm should take them into account in our score, so that afterwards classifier can pick such a cases.

(2) *Difference feature:* Different words make the subject of each sentence different.
Idea for solution: create a score (algorithm) which gets the main subject of the sentence, and checks if both of the sentences are of the same subject, so that afterwards classifier can pick such a cases.

Table 8. Different words make the subject of each sentence different

Sentence pairs and explanations
Agent: again the RFU delays the salary of Fabio Capello./ Агент: РФС вновь задерживает зарплату Фабио Капелло.
Mass media: Agent of Fabio Capello is threatening to sue the RFU./ СМИ: Агент Фабио Капелло грозится подать в суд на РФС.
in first sentence: RFU delays salary in second sentence: Fabio Capello is threatening to sue.
The results of the elections in the UK has strengthened the pound./ Результаты выборов в Великобритании укрепили фунт.
Became known the final results of the elections in the UK./ Стали известны окончательные результаты выборов в Великобритании.
in first sentence: results strengthened the pound in second sentence: results became known

9.2 False Negatives: Mistakenly Predicted as 'Non-paraphrase'

Rule for Such Pairs, with Corresponding Examples. Such pairs of sentences typically contain for the most part *different words* but the meaning of the whole sentence is the the same or closely related (since I have only one paraphrase class for second task, I can combine closely related sentences to paraphrase class): Table 9.

Table 9. Sentence pairs

Apakan called on the parties of the Ukrainian conflict to observe a truce on 9 may./ Апакан призвал стороны украинского конфликта соблюдать перемирие 9 мая.
The OSCE has called for peace in Ukraine on May 9./ В ОБСЕ призвали к миру на Украине 9 Мая.
Media: the United States are planning to deploy Osprey convertiplane in the suburbs of Tokyo./ СМИ: США планируют разместить конвертопланы Osprey в пригороде Токио.
The Pentagon has decided to place in the suburbs of Tokyo ten convertiplanes./ Пентагон решил разместить в пригороде Токио десять конвертопланов
Media: police of United States of America is looking for unknowns who have stolen the clothes of Demi Moore./ СМИ: полиция США ищет неизвестных, укравших одежду Деми Мур.
Demi Moores' clothes for 200 thousand dollars were stolen./ У Деми Мур украли одежду на 200 тысяч долларов.

Idea How to Solve Such Cases. Pay attention that the word Apakan in the *first pair* is the name of the current Chief Monitor of the OSCE Special Monitoring Mission to Ukraine. So if I would have knowledge base, which associated Apakan with OSCE - and gave that knowledge base to those scores (algorithms), that would have helped to identify the similarity of these two sentences.

The same goes with the *second pair* since Pentagon is United States *state level* organization.

In the *third pair* I need sophisticated understanding since ‘clothes of Demi Moore’ mentioned in first sentence can be expensive enough, since she is a celebrity, and can afford to buy something expensive. So because in the second sentence is mentioned the price of the clothes, and due to this amount of money (‘200 thousand dollars’) - we understand that is being talked about expensive clothes - we can conclude that the subject of both sentences in this pair is the same (e.g. it is about ‘expensive clothes that were stolen’).

Such a sophisticated AI can be created, but it remains an open challenge.

10 Conclusions and Future Work

In this work I address the paraphrase classification problem by combining Machine Translation and using an ensemble of semantic similarity algorithms. The resulting F1 score and Accuracy metrics have shown the effectiveness of such an algorithm. I achieved recognition of **64.14%** in Accuracy and **62.46%** in F1 score for the First Task (3-way classification), and **81.41%** in Accuracy and **78.51%** in F1 score for the Second Task (two-way classification). Results on both tasks are significantly better than the baseline BLEU algorithm (ran on Russian source, without translation).

I completed ablation test to detect which of those algorithms gave more effect in gaining the correct answer. I observe that the best toolkit is SEMILAR. Interesting enough, python difflib showed pretty good result for just a regular python library, which results in only about 5% less than our winner - SEMILAR. The best score of all used, for given corpus, is greedyComparerWNLin, which is a part of the SEMILAR toolkit. The best translation engines, for our corpus in Russian, are Yandex and Google.

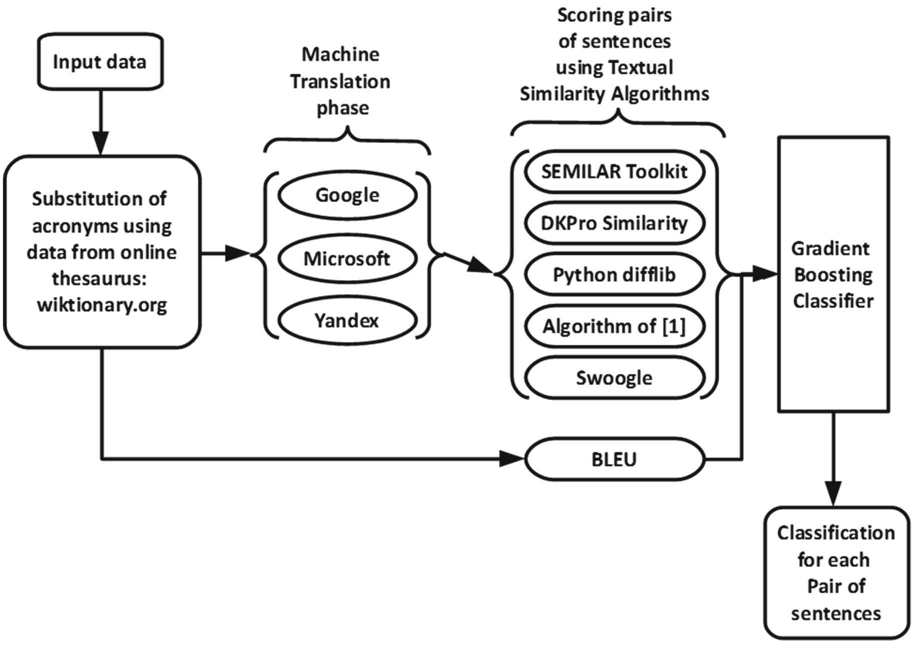
I used most popular semantic similarity algorithms (toolkits) however more algorithms are available. In future work I intend to use additional algorithms to improve paraphrase recognition. Additionally I plan to develop scores which will allow us to cover the cases mentioned in the error analysis section, to let the classifier to include these cases into the correct classes.

Although the corpus was in Russian and I translated it to English using automated translation engines (Google, Microsoft and Yandex), and semantic similarity algorithms I launched on translations which were not 100% correct in the suitability of translated words and syntactic structure, since these translations engines can give only approximation to the most fitted translation. So F1 score and Accuracy could be higher if the corpus was originally in English.

11 Repository

Python code sources are available on:
https://github.com/dmikrav/paraphraser.ainlconf.2016
Web-site is: https://dmikrav.github.io/paraphraser.ainlconf.2016/.

A Appendix: Algorithm Data-Flow


Input data
Substitution of acronyms using data from online thesaurus: wiktionary.org
Machine Translation phase
Google
Microsoft
Yandex
Scoring pairs of sentences using Textual Similarity Algorithms
SEMILAR Toolkit
DKPro Similarity
Python difflib
Algorithm of [1]
Swoogle
BLEU
Gradient Boosting Classifier
Classification for each Pair of sentences

B Appendix: Ablation Test Tables

Table 10. Each toolkit launched separately

Toolkit	Accuracy	F1 score
SEMILAR	**80.13**	**77.02**
DKPro Similarity	79.52	75.78
Python difflib	75.92	71.36
Algorithm of [1]	78.76	75.02
Swoogle	78.94	75.03

Table 11. Combinations by two toolkits

Toolkits	Accuracy	F1 score
SEMILAR + DKPro similarity	**80.86**	**77.89**
SEMILAR + Python difflib	80.37	77.34
SEMILAR + Algorithm of [1]	80.18	77.10
SEMILAR + Swoogle	80.17	77.05
DKPro similarity + Python difflib	79.59	75.95
DKPro similarity + Algorithm of [1]	80.04	76.62
DKPro similarity + Swoogle	80.26	76.87
Python difflib + Algorithm of [1]	79.09	75.44
Python difflib + Swoogle	79.42	75.69
Algorithm of [1] + Swoogle	80.10	76.66

Table 12. Combinations by three toolkits

Toolkits	Accuracy	F1 score
SEMILAR + DKPro similarity + Python difflib	80.66	77.66
SEMILAR + DKPro similarity + Algorithm of [1]	80.55	77.59
SEMILAR + DKPro similarity + Swoogle	**80.89**	**77.89**
SEMILAR + Python difflib + Algorithm of [1]	80.43	77.45
SEMILAR + Python difflib + Swoogle	80.61	77.66
SEMILAR + Algorithm of [1] + Swoogle	80.73	77.77
DKPro similarity + Python difflib + Algorithm of [1]	80.37	77.14
DKPro similarity + Python difflib + Swoogle	79.93	76.55
Python difflib + Algorithm of [1] + Swoogle	80.03	76.60

Table 13. Combinations by four and five toolkits

Toolkits	Accuracy	F1 score
SEMILAR + DKPro similarity + Python difflib + Algorithm of [1]	80.94	77.99
SEMILAR + DKPro similarity + Python difflib + Swoogle	**81.36**	**78.39**
SEMILAR + Python difflib + Algorithm of [1] + Swoogle	80.75	77.79
DKPro similarity + Python difflib + Algorithm of [1] + Swoogle	80.72	77.56
All five toolkits together	**81.41**	**78.51**

References

1. Li, Y., McLean, D., Bandar, Z., O'Shea, J., Crockett, K.: Sentence similarity based on semantic nets and corpus statistics. IEEE Trans. Knowl. Data Eng. **18**, 1138–1150 (2006)
2. Chen, B., Cherry, C.: A systematic comparison of smoothing techniques for sentence-level BLEU. In: WMT@ACL, pp. 362–367 (2014)
3. Ding, L., Finin, T., Joshi A., Peng, Y., Cost, S., Sachs, J., Pan, R., Reddivari, P., Doshi, V.: Swoogle: a semantic web search and metadata engine. In: CIKM, pp. 652–659 (2004)
4. Rus, V., Lintean, M.: A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In: BEA@NAACL-HLT, pp. 157–162 (2012)
5. Ştefănescu, D., Banjade, R., Rus, V.: Latent semantic analysis models on Wikipedia and TASA. In: LREC, pp. 1417–1422 (2014)
6. Lintean, M., Rus, V.: Paraphrase identification using weighted dependencies and word semantics. Informatica (Slovenia) **34**, 19–28 (2010)
7. Rus, V., Lintean, M., Banjade, R., Niraula, N., Stefanescu, D.: SEMILAR: the semantic similarity toolkit. In: ACL (Conference System Demonstrations), pp. 163–168 (2013)
8. Banjade, R., Niraula, N., Maharjan, N., Rus, V., Stefanescu, D., Lintean, M., Gautam, D.: NeRoSim: a system for measuring and interpreting semantic textual similarity. In: Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT, pp. 164–171 (2015)
9. Bar, D., Zesch, T., Gurevych, I.: DKPro similarity: an open source framework for text similarity. In: ACL (Conference System Demonstrations), pp. 121–126 (2013)
10. Pronoza, E., Yagunova, E.: Low-level features for paraphrase identification. In: Sidorov, G., Galicia-Haro, S.N. (eds.) MICAI 2015. LNCS (LNAI), vol. 9413, pp. 59–71. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27060-9_5
11. Ganitkevitch, J., Van Durme, B., Callison-Burch, C.: PPDB: the paraphrase database. In: HLT-NAACL, The Association for Computational Linguistics, pp. 758–764 (2013)
12. Dolan, B., Brockett, C., Quirk, C.: Microsoft Research Paraphrase Corpus (2005)
13. Ji, Y., Eisenstein, J.: Discriminative improvements to distributional sentence similarity. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 891–896 (2013)
14. Socher, R., Huang, E., Pennington, J., Ng, A., Manning, C.: Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems, pp. 801–809 (2011)
15. Yin, W., Schutze, H.: Convolutional neural network for paraphrase identification. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, HLT-NAACL, pp. 901–911 (2015)
16. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML), pp. 296–304 (1998)
17. Dagan, I., Dolan, B., Magnini, B., Roth, D.: Recognizing textual entailment: rational, evaluation and approaches - Erratum, In: Natural Language Engineering, vol. 16 (2010)

18. Madnani, N., Tetreault, J., Chodorow, M.: Re-examining machine translation metrics for paraphrase identification. In: Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, pp. 182–190 (2012)
19. Chitra, A., Rajkumar, A.: Plagiarism detection using machine learning-based paraphrase recognizer. J. Intell. Syst. **25**, 351–359 (2016)
20. Dey, K., Shrivastava, R., Kaushik, S.: A paraphrase and semantic similarity detection system for user generated short-text content on Microblogs. In: COLING, 26th International Conference on Computational Linguistics, Proceedings of the Conference, pp. 2880–2890 (2016)
21. Pivovarova, L., Pronoza, P., Yagunova, E., Pronoza, A.: ParaPhraser: Russian paraphrase corpus and shared task. In: Filchenkov, A., et al. (eds.) AINL 2017. CCIS, vol. 789, pp. 211–225. Springer, Cham (2018)