Определение семантической близости текстов с использованием инструмента DKPro Similarity

А.В. Крюкова

Санкт-Петербургский государственный университет

krukova.ann@gmail.com

Аннотация

В данной работе рассматривается проблема оценки семантической близости текстов на русском языке. Мы описываем преимущества использования открытой компьютерной платформы DKPro Similarity для решения этой проблемы, сосредоточив внимание на строковых метриках оценки близости текстов. Эксперименты проводятся на материале тестовой выборки, включающей сходные фрагменты художественных, научных и новостных текстов. Мы используем несколько представленных в платформе DKPro Similarity метрик и передаем полученные значения алгоритмам машинного обучения в качестве признаков. Результаты работы показывают, что простые строковые метрики позволяют достичь высоких результатов при определении отнесенности текстов к одной группе с помощью линейных моделей. В исследовании также предлагается метод оценки релевантности метрик для конкретных задач.

Ключевые слова: семантическая близость, метрики семантической близости, DKPro Similarity, машинное обучение, компьютерная лингвистика.

1. Введение

Оценка семантической близости текстов является неотъемлемой составляющей многих задач современной компьютерной лингвистики, среди которых создание и функционирование информационно-поисковых систем, вопросно-ответных систем, систем автоматического реферирования, классификации текстов, определения тематики текстов, перефразирования, разрешения лексической неоднозначности и др. До сих пор разработка и тестирование алгоритмов и метрик для оценки семантической близости текстов проводились в основном применительно к материалу английского языка. Это можно проиллюстрировать классом компьютерных инструментов, созданных

для решения данной задачи: ср. WordNet::Similarity[1], Alchemy API[2] и ряд других. Об успехах в этой области также свидетельствуют результаты соревнований SemEval[3] на специальной дорожке Semantic Textual Similarity. Необходимость подобных исследований для русского языка обусловлена востребованностью ожидаемых результатов в компьютерной лингвистике. В частности, метрики семантической близости текстов и определение семантических отношений между словами могут использоваться при создании инструментов автоматического понимания текстов.

В настоящий момент есть прогресс в области автоматической оценки семантической близости на уровне слов (ср. данные RUSSE[4] [1]), однако задача определения близости текстов не подвергалась тщательному изучению: акцент делается не на количественных данных о схожести текстов, а на результатах кластеризации или классификации большого числа документов в корпусе (например, когда нужно определить тематику корпуса или назначить рубрики для отдельных его сегментов). Наше исследование призвано восполнить существующий пробел[5].

Итак, в данной работе мы решаем задачу оценки семантической близости текстов на русском языке средствами открытой и свободно распространяемой компьютерной платформы DKPro Similarity, что предполагает изучение возможностей данной платформы, адаптацию инструмента для работы с русским языком, эксперименты на текстовом материале с использованием различных метрик семантической близости.

2. Компьютерный инструмент DKPro Similarity

Компьютерный инструмент DKPro Similarity[6] разработан в Дармштадском Технологическом Университете (TU Darmstadt)[7] исследовательской группой [2]. Эта платформа была создана как дополнение DKPro Core, набора компонентов ПО для обработки естественного языка; инструмент поддерживается на языках Java, Jython и Groovy. Преимуществами данной платформы являются ее открытый характер, реализация множества существующих метрик близости текстов с использованием стандартизованного способа их вызова, а также возможность разрабатывать собственные метрики на основе уже существующих. DKPro Similarity включает в себя различные классы метрик близости текстов: структурные, стилистические, строковые, семантические и

[1] URL: http://wn-similarity.sourceforge.net/

[2] URL: http://www.ibm.com/watson/developercloud/alchemy-language.html

[3] URL: http://aclweb.org/aclwiki/index.php?title=SemEval_Portal

[4] URL: http://russe.nlpub.ru/

[5] Предыдущий этап работы см. на сайте конференции Диалог-2017
URL: http://www.dialog-21.ru/media/3985/kriukova.pdf

[6] URL: https://dkpro.github.io/dkpro-similarity/

[7] URL: https://www.ukp.tu-darmstadt.de/software/dkpro-core/

фонетические[8]. В нашем исследовании мы опираемся на строковые метрики, не зависящие от языка обрабатываемого текста.

3. Лингвистические данные

Мы сформировали экспериментальную выборку текстов таким образом, что в него вошли тексты, результаты вычисления близости которых можно было адекватно оценить (подробнее об этом см. раздел 4.2), так как для русского языка нет «золотого стандарта» для оценки семантической близости текстов, т.е. корпуса, в котором пары текстов были бы снабжены экспертными оценками их сходства[9]. Таким образом, материалом исследования послужили следующие группы текстов:

- аннотации научных статей из корпуса по корпусной лингвистике кафедры математической лингвистики СПбГУ;
- сообщения из сегментов «life» и «news» новостного корпуса кафедры математической лингвистики СПбГУ;
- три перевода на русский язык романа В. Набокова «Пнин» (а именно, переводы Г.А. Барабтарло, С.Б. Ильина, Б.М. Носика);
- заголовки новостных статей из корпуса парафразов в проекте ParaPhraser.ru[10].

Из каждой группы случайным образом было выбрано несколько текстов для дальнейшей работы: пять аннотаций; по пять сообщений из двух частей новостного корпуса; пять соответствующих друг другу отрывков из трех переводов; а также по пять пар парафразов из каждой группы в корпусе (преимуществом является то, что в этом корпусе каждой паре предложений соответствует экспертная оценка того, в какой мере они действительно являются парафразами: «-1»—предложения на разные темы, «0» — предложения на одну тему, но есть изменения смысла, «1» — абсолютные парафразы). В табл. 1 можно найти информацию о длине используемых текстов.

Таблица 1. Средняя, максимальная и минимальная длина текстов (в символах)

	Длина		
	Средняя	**MAX**	**MIN**
Пнин	1607	2311	984
Аннотации	560	650	440
News	1275	1525	1153
Life	1163	1853	687
Парафразы	62	74	44

[8] Полный список реализованных в DKPro метрик близости текстов можно найти в хранилище на GitHub: https://github.com/dkpro/dkpro-similarity. Однако следует учитывать, что многие из них не применимы к русскому языку.

[9] См. подобные ресурсы для английского – URL: https://www.aclweb.org/aclwiki/index.php?title=Similarity_(State_of_the_art)

[10] URL: http://www.paraphraser.ru/about/

Перед вычислением семантической близости текстов была проведена их обработка: удаление знаков препинания и лемматизация с использованием библиотеки PyMorphy2[11] [3].

4. Ход экспериментов

4.1. Используемые метрики семантической близости текстов

В DKPro Similarity реализовано более 15 различных строковых метрик близости, из которых в нашем исследовании мы использовали семь наиболее обсуждаемых, ср. [4] [5] [6] [7]:

1. Word N-Gram Containment Measure — документы разбиваются на *n*-граммы, и «мера включения» выражается следующей формулой: $Cn(A,B)= \frac{|S(A,n) \cap S(B,n)|}{|S(A,n)|}$, где $S(A,n)$ и $S(B,n)$ — это множество *n*-грамм в документах *A* и *B* соответственно (см. [8]);
2. Word N-Gram Jaccard Measure — документы разбиваются на n-граммы, и для них вычисляется коэффициент Жаккара: отношение количества общих n-грамм к количеству *n*-грамм в целом (см. [9]);

В обеих метриках с *n*-граммами мы используем параметр $n = 2$.

3. Levenshtein Comparator — вычисляется минимальное количество операций вставки или удаления одного символа или замены его на другой, необходимых для преобразования одной строки в другую (см. [10]);
4. Longest Common Subsequence Comparator — самая длинная общая подпоследовательность вычисляется через нахождение наибольшего количества операций вставки или удаления символов (для строк, оставшихся после удаления общей подпоследовательности); затем производится нормализация: $1 - \frac{|A|+|B|-2*|LCS(A,B)|}{|A|+|B|}$, где $|A|$ и $|B|$ — количество символов в документах *A* и *B* соответственно (см. [11]);
5. Greedy String Tiling — алгоритм ищет такое разбиение документов *A* и *B* на непересекающиеся друг с другом одинаковые цепочки (tiles), при котором ими окажется покрытым наибольшее число токенов в документах (см. [12]); на вход алгоритм принимает минимальную длину цепочек для поиска (по умолчанию она равна трем); нормализуется результат следующим образом: количество «покрытых» токенов делится на количество токенов во втором документе: $GST(A,B)= \frac{\sum_{i \in tiles} length_i}{|B|}$ (см. [13]);
6. Longest Common Substring Comparator — самая длинная подстрока вычисляется с помощью общего для двух строк дерева суффиксов; полученное значение нормализуется так же, как и в метрике с общей подпоследовательностью;
7. Cosine Similarity — строятся векторные представления сравниваемых текстов, рассчитывается косинус угла между векторами; по умолчанию

[11] URL: https://github.com/kmike/pymorphy2

веса термов в документе равны частоте их встречаемости, а норма векторов вычисляется стандартно, как корень из суммы квадратов их координат (см. [10]).

Следует отметить, что значение всех из них принадлежит отрезку [0, 1], кроме расстояния Левенштейна, которое, наоборот, равно нулю, если два текста идентичны, и тем больше, чем больше в них различий в символах, причем это число ограничено сверху только длиной большего текста. Также важной деталью является то, что значение двух из данных метрик — Word N-Gram Containment Measure и Greedy String Tiling — зависит от порядка, в котором документы сравниваются друг с другом: в знаменателе формулы стоит число, связанное только с одним из текстов (количество *n*-грамм в первом документе и длина второго документа соответственно). В связи с этим при использовании этих метрик мы вычисляли оба значения.

Для текстов из каждой группы (см. раздел 3) было вычислено девять значений близости, в результате чего мы получили несколько таблиц с результатами: пять для каждой группы текстов и одну, в которой сравнивались тексты из разных групп. В табл. 2 можно увидеть, как выглядели значения всех метрик близости для сравнений нескольких пар аннотаций.

Таблица 2. Значения метрик близости для нескольких первых аннотаций

	1+2[12]	1+3	1+4
N-Gram Containment	0,048	0,081	0,032
	0,045	0,104	0,038
N-Gram Jaccard	0,024	0,048	0,018
Levenstein	476	374	425
Cosine	0,254	0,365	0,286

	1+2	1+3	1+4
Longest Subsequence	0,443	0,483	0,432
Greedy String Tiling	0,417	0,419	0,457
	0,347	0,522	0,478
Longest Substring	0,023	0,049	0,03

4.2. Оценка результатов близости текстов

Как мы уже говорили, «золотого стандарта» для задачи определения схожести текстов на русском языке не существует, поэтому мы выработали собственные способы оценки полученных результатов. Напомним, что в корпусе парафразов предложения уже были оценены вручную (см. раздел 3). Мы решили следовать такому же методу, но он подходит только для текстов, которые изначально близки друг к другу: из наших материалов этому параметру соответствуют отрывки переводов романа «Пнин». Каждый из них мы разбили на небольшие фрагменты (по одному-двум предложениям), соответствующие друг другу в разных переводах. В результате каждый из пяти изначальных отрывков был представлен несколькими текстовыми документами, в которых находилось три маленьких отрывка. Был проведен эксперимент с участием

[12] Здесь и в последующих таблицах цифры в названиях столбцов или строк — это условные обозначения (порядковые номера) сравниваемых текстов.

информантов — экспертов (студентов кафедры математической лингвистики). Мы попросили их оценить попарное сходство фрагментов. Семантическая близость целостных текстов определялась через средние оценки близости их отрывков. Каждое значение оценивалось двумя участниками, и сравнение проводилось отдельно по двум критериям:

1) Смысловой критерий: насколько тексты похожи по смыслу (используется шкала «0–1–2», где «2» — сильная степень схожести, «1» – средняя степень, «0» — небольшая степень схожести);
2) Формальный критерий: насколько близость текстов определяется входящими в их состав словами (также используется шкала «0–1–2», но при оценке предлагалось учитывать критерии, схожие с критериями автоматического распознавания парафраз в проекте ParaPhraser.ru):
    a) наличие одинаковых слов;
    b) наличие синонимов / транспозиции / общих корней.

В результате для каждой пары сравниваемых текстов было получено два значения от «0» до «2»: одно выражает близость текстов со смысловой точки зрения, другое — с формальной, и именно они использовались в машинном обучении (см. раздел 4.3.). Для оценки согласованности ответов участников эксперимента мы использовали взвешенную Каппу Коэна (Weighted Cohen's Kappa), присваивая вес 0,1 неодинаковым ответам, отличающимся друг от друга на единицу (то есть оценкам «0» и «1» или «1» и «2»), и вес 10 — тем случаям, когда участники поставили оценки «0» и «2» одному отрывку. Показатель согласованности оказался равен 0,68.

Однако для других текстов из экспериментальной выборки подобный критерий не применим, так как они изначально не объединены общей темой. Поэтому мы следовали следующей стратегии: исходя из того, что тексты из одной группы похожи друг на друга больше, чем на тексты из других групп, каждой паре документов мы поставили оценку «1», если они принадлежат одной группе, и «0», если они относятся к разным. После этого шага количество наборов данных увеличилось:

- результаты для текстов из корпуса новостей были разделены на три набора: два, состоящих из значений близости текстов внутри подгрупп «news» и «life» отдельно, и один, включающий сравнения текстов как внутри этих подгрупп, так и между собой;
- из группы с текстами романа «Пнин» было так же составлено два набора: один включает только сравнения переводов одинаковых фрагментов, другой — сравнения любых текстов из романа;

(эти разделения обусловлены тем, рассматриваем ли мы все тексты из корпуса новостей или романа «Пнин» как относящиеся к одной группе или к разным);

- также мы создали еще один набор данных, в котором близость парафраз оценивалась бинарно: «0», если в корпусе стояло значение «-1», т.е. если предложения друг с другом никак не связаны, и «1», если в корпусе были указаны оценки «0» или «1».

4.3. Обучение

На данном этапе у нас было одиннадцать наборов данных, объектами в которых выступали пары текстов, а их признаками — значения мер близости. К признакам мы также добавили длину обоих текстов, так как от них зависит значение некоторых метрик (см. раздел 4.1.). Таким образом, каждый объект характеризовался одиннадцатью признаками, а также эталонным значением: для семи датасетов это было значение «1», для одного (не-бинарной оценки парафраз) — «-1», «0» или «1», для двух (смысловой и формальной близости отрывков из романа «Пнин») — вещественное значение от 0 до 2, и для одного (текстов из разных групп) — значение «0». Каждый датасет со значением целевого признака «1» был объединен с датасетом с «0», значения признаков отмасштабированы.

В результате каждый набор данных состоял в среднем из 35 объектов. Так как для оценки результатов мы используем трехкратную кросс-валидацию, каждый раз объем обучающей выборки составлял примерно 66% (23 объекта), а тестовой, соответственно, 33% (12 объектов). Машинное обучение производилось на языке программирования Python, с использованием библиотеки Scikit-learn[13]. В ней можно найти реализацию методов, о которых речь пойдет дальше.

Для задач классификации эксперименты проводились с несколькими линейными моделями: Logistic Regression, Ridge Classifier, SGD Classifier, Passive Aggressive Classifier, Perceptron. Различие в методах их работы для нашего исследования оказалось несущественным, мы использовали общий принцип работы линейных моделей. Каждому признаку объектов они присваивают определенный коэффициент — его вес, который умножается на значение признака у каждого конкретного объекта, потом полученные значения складываются, и в зависимости от результата объект относится к тому или иному классу: к классу «1», если результат положительный, и к классу «0»,если он отрицательный.

Таблица 3. Лучшие результаты для каждого набора текстов

All Pnin	**Pnin binar**	**Paraphrases**	**Paraphrases binar**
0,87	1	0,861	1
Logistic Regression (L1 regression)	Logistic Regression	Logistic Regression (+ Grid Search)	Logistic Regression

All News	**News**	**Life**	**Annotations**
0,841	0,872	0,631	1
Logistic Regression (L1 regression)	Ridge Classifier	Passive Aggressive Classifier	Logistic Regression

[13] URL: http://scikit-learn.org/stable/

В результате для каждого набора данных выбиралась одна модель, показавшая наилучший результат при трехкратной кросс-валидации с оценкой результатов по F-мере (см. Таблицу 3). Поясним некоторые обозначения в таблице. Если название метода не снабжено дополнительными комментариями, использовалась его реализация с параметрами по умолчанию. «Grid Search» — подбор наилучшей комбинации параметров из предложенных пользователем. «L1 regression» — это использование L1-регуляризации (регуляризации Лассо) вместо L2-регуляризации, которая применяется по умолчанию.

Для двух задач, где ответом должно было быть вещественное число (датасеты с проставленной вручную оценкой близости фрагментов из романа «Пнин»), также использовались линейные модели, но они не показали точных результатов. Меньших значений средней абсолютной ошибки (mean absolute error) удалось достичь, используя Random Forest Regressor: для оценок по смыслу ошибка оказалась равна 0,208, а для формальных оценок — 0,188.

4.4. Выводы

Как мы видим, классификаторы, обученные даже на таких простых признаках, как значения строковых метрик близости текстов, показывают неплохие результаты. В связи с этим мы предлагаем использовать веса, назначенные признакам линейными моделями, в качестве способа оценки строковых метрик для конкретной задачи. Для этой цели была составлена таблица со значениями весов лучших моделей (см. табл. 4: в ней выделены наибольшие веса для каждого датасета).

Для каждой метрики мы вычислили среднее значение весов по модулю (при этом мы не учитывали веса для датасетов «All News» и «All Pnin»[14], так как из-за использования регуляризации Лассо (L1) многие признаки имеют нулевой вес, а другие, наоборот, значительно бóльшие веса по сравнению с остальными). Вычисления показали, что наибольшие веса у N-Gram Containment Measure и Cosine Similarity. Таким образом, именно они наиболее подходят для поставленной задачи.

5. Заключение

Итак, мы показали, что даже строковые метрики оценки близости текстов, которые, как обычно считается, дают хорошие результаты только для очень схожих по наборам слов текстов, в данном случае позволяют линейным моделям достаточно хорошо работать при классификации текстов. В дальнейшем также планируется провести эксперименты с семантическими метриками близости, опирающимися на внешние источники знаний (например, на «Википедию»: см. ESA [14]), и сравнить результаты, которые будут получены, с результатами данного исследования. Также планируется расширить языковой материал и, возможно, использовать более сложные модели машинного обучения для получения лучших результатов.

[14] См. табл. 4.

Таблица 4. Веса признаков для всех наборов данных

	Pnin binar	**Paraphrases binar**	**Paraphrases**[15]		
N Gram Containment1	0,585	0,393	0,025	**-1,458**	**1,514**
N Gram Containment2	**0,612**	0,37	-0,062	-0,255	0,189
N Gram Jaccard	0,573	0,357	0,067	-0,787	0,621
Levenstein	-0,461	-0,426	**-0,922**	0,833	-0,087
Longest Subsequence	**0,607**	0,434	-0,216	0,69	-0,971
Greedy String1	0,422	**0,616**	**-1,194**	-0,011	**1,511**
Greedy String2	0,489	**0,738**	-0,706	0,897	-0,177
Longest Substring	0,078	0,44	-0,401	0,067	0,348
Cosine	0,517	0,585	-0,876	**1,46**	-0,962
Length1	0,105	-0,433	-0,574	0,557	0,054
Length2	0,054	-0,473	-0,825	1,069	-0,356

	News	**Life**	**Annotations**	**All Pnin**	**All News**
N Gram Containment1	0,163	0,48	0,786	0	**6,648**
N Gram Containment2	-0,338	0,069	**0,695**	0	**-6,396**
N Gram Jaccard	-0,015	0,247	**0,75**	0	0
Levenstein	0,142	-0,394	-0,54	4,498	-0,037
Longest Subsequence	0,085	**-0,744**	0,447	0	0
Greedy String1	**0,534**	-0,15	0,06	3,1	1,008
Greedy String2	**0,439**	-0,425	0,018	**7,708**	2,107
Longest Substring	0,004	0,241	0,061	0	1,882
Cosine	-0,086	0,245	-0,058	1,925	-0,434
Length1	0,158	**-0,614**	-0,427	**-6,038**	0
Length2	0,117	-0,03	-0,578	-1,011	0

[15] Для данного набора данных (парафразы с оценками «1», «0», «-1») классов не два, как в остальных датасетах, а три, и классификатор работает по принципу «один против всех» (One VS All), отделяя каждый из них от двух остальных; именно поэтому здесь три набора весов.

Литература

[1] Panchenko A. et al. RUSSE: The First Workshop on Russian Semantic Similarity // Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference «Dialogue 2015». M., 2015. Pp. 89–105.

[2] Bär D., Zesch T., Gurevych I. DKPro Similarity: An Open Source Framework for Text Similarity // Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations. 2013. Pp. 121–126.

[3] Korobov M. Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts. Springer, 2015. Pp. 320 – 332.

[4] Mihalcea R. et al. Corpus-based and Knowledge-based Measures of Text Semantic Similarity // Proceedings of the 21st National Conference on Artificial Intelligence. 2006. Vol. 1. Pp. 775–780.

[5] Bär D. et al. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures // SemEval–2012 Proceedings of the First Joint Conference on Lexical and Computational Semantics: Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. 2012. Pp. 435–440.

[6] Šarić F. et al. TakeLab: Systems for Measuring Semantic Text Similarity // SemEval–2012 Proceedings of the First Joint Conference on Lexical and Computational Semantics: Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. 2012. Pp. 441–448.

[7] Bär D., Zesch T., Gurevych I. Composing Measures for Computing Text Similarity [Technical Report]. 2015. URL: http://tuprints.ulb.tu-darmstadt.de/4342/1/TUD-CS-2015-0017.pdf.

[8] Broder A.Z. On the resemblance and containment of documents // Proceedings of the Compression and Complexity of Sequences. 1997. Pp. 21–29.

[9] Lyon C., Barrett R., Malcolm J. A theoretical basis to the automated detection of copying // Plagiarism: Prevention, Practice and Policies Conference. 2004.

[10] Manning C.D. et al. Introduction to Information Retrieval. Cambridge University Press, 2008.

[11] Clough P., Stevenson M. Special Issue on Plagiarism and Authorship Analysis // Language Resources and Evaluation. 2011. Vol. 45. № 1. Pp. 5–24.

[12] Wise M.J. Yap3: Improved detection of similarities in computer programs and other texts // Proceedings of SIGCSE–1996. 1996. Pp. 130–134.

[13] Clough P. et al. METER: MEasuring TExt Reuse // Proceedings of the 40th Annual Meeting of the ACL. 2002. Pp. 152–159.

[14] Gabrilovich E., Markovitch S. Wikipedia-based Semantic Interpretation for Natural Language Processing. // Journal of Artificial Intelligence Research. Vol. 34. 2009. Pp. 443–498.

Computing semantic similarity of Russian texts by means of DKPro Similarity tool

A. Kriukova

Saint-Petersburg State University

This paper looks into a problem of computing semantic similarity of texts in Russian. In course of experiments we employ an open-source framework DKPro Similarity, and describe its advantages for this purpose. Our attention is focused on string metrics of computing text similarity. Experiments are carried out on test samples including similar extracts from fiction, research, and news texts. For pairs of texts we use several string-based similarity metrics, implemented in DKPro Similarity, and pass the computed values as features for machine learning algorithms. We also present a method of evaluating similarity measures' relevance for particular purposes. Results of the research prove that simple string-based metrics contribute to performance of linear models while trying to identify whether texts belong to the same group – with average F-measure value 0,88 for eight datasets. In future we also plan to use semantic text similarity measures which make use of external sources of knowledge, e.g. Wikipedia, and employ more sophisticated machine learning algorithms to improve the performance in some difficult cases.

Keywords: semantic similarity, semantic similarity measures, DKPro Similarity, machine learning, computational linguistics.