

Пояснительная записка

к дипломному проекту на тему:

**«Анализ статьи “Обучение на основе метрики семантического расстояния для задачи детекции лексико-семантических изменений” (A semantic distance metric learning approach for lexical semantic change detection)»**

Автор: Чумаченко Алексей Андреевич

Группа: DSU-34

**Оглавление**

1. Анализ теоретической части статьи и применяемых подходов
2. Анализ данных для реализации
3. Методика повторения эксперимента
4. Итоги обучения модели и полученные результаты
5. Выводы и заключение
6. Список источников
   * + 1. **Анализ теоретической части статьи и применяемых подходов**

Для дипломной работы была выбрана статья авторства Таичи Аиды (Taichi Aida) и Данушки Боллегалы (Danushka Bollegala) “Обучение на основе метрики семантического расстояния для задачи детекции лексико-семантических изменений” (A semantic distance metric learning approach for lexical semantic change detection)1. В данной статье авторы рассматривают проблему изменения значений слов в зависимости от контекста, что может усложнять обработку естественного языка (natural language processing) и приводить к снижению качества предсказаний. В данном конкретном случае авторы рассматривают задачу детекции лексико-семантических изменений (SCD), в которой необходимо определить, меняет ли целевое слово свое значение в двух корпусах текстов. Распознавание слов с меняющимися значениями позволяет эффективно дообучать (fine-tuning) большие языковые модели (LLM), чтобы использовать только те значения, которые подходят для конкретной специфичной задачи.

Целями дипломного проекта являются:

1. Познакомиться с задачей Semantic Change Detection
2. Повторить эксперимент, описанный в статье (в соответствии с имеющимися возможностями);
3. Сравнить полученные результаты с описанными в статье;
4. Проверить эффективность предложенного метода.

Авторы применили двухэтапный подход для решения этой задачи. На первом этапе используется датасет WiC (слова в контексте) для получения эмбедингов набора слов в зависимости от их значений в контексте. Для преобразования слов в эмбединги применяется Siamese bi-encode, в основе которого лежит модель XLM-RoBERTa-Large на архитектуре Bert (XL-LEXEME)2. В общем случае такой энкодер состоит принимает на вход две последовательности слов и используя Siamese Network (Сиамская нейронная сеть) преобразует их в два разных векторных представления. Каждая последовательность токенизируется и обрезается до заданной максимальной длинны. Вектор последовательности вычисляется как сумма закодированных слов, входящих в состав последовательности. Энкодер натренирован на минимизацию ошибки следующего вида:

ℓ = ½ · [y · δ2 + (1 − y) · max(0, m − δ)2] (1),

где m = 0.5, а δ – косинусное расстояние между закодированными представлениями входных последовательностей. В данной работе вместо последовательности кодируется целевое слово в двух разных контекстах и в уравнении (1) δ – косинусное расстояние между закодированными представлениями одного и того же слова в разных контекстах. Таким образом, на первом этапе авторы статьи получают набор эмбедингов, каждый из которых представляет отношение значений одного и того же слова, но в разных контекстах.

На втором этапе происходит непосредственно обучение метрики расстояния на основе значения (sense-aware distance metric). Математически данная метрика описывается следующей формулой:

h(w1, w2; A) = (w1 − w2)T · A · (w1 − w2) (2),

где w1 = f (w, s1; θ) и w2 = f (w, s2; θ) – эмбединги целевого слова для двух предложений, то есть основанные на значении слова в каждом конкретном предложении, а A – матрица Махаланобиса, которая определяет метрику расстояния. Тренировочный датасет WiC используется именно для обучения матрицы Махаланобиса. Для каждого слова есть два предложения и бинарная метка: 1 (или True) – слово имеет одно и то же значение в обоих предложениях, 0 (или False) – значения слова отличаются. Для обучения матрицы A используется алгоритм ITML (Information Theoretic Metric Learning)3. Для реализации алгоритма инициализируются следующие параметры:

* Матрица X – матрица эмбедингов размерности dx2n, где d – размерность вектора эмбединга, n – количество примеров в датасете
* y – метки совпадения/несовпадения значений
* l, u – верхнее и нижнее пороговые значения для метрики дистанции
* A0 – начальное значение матрицы Махаланобиса (на первом шаге нулевая)
* γ – параметр регуляризации.

Описание алгоритма ITML приведено ниже:

**Input**: input matrix **X**, labels **y**, distance thresholds **[l, u]**, input Mahalanobis matrix **A0**, slack parameter **γ**

**Output**: Mahalanobis matrix A

**1:** # Initialise A, λ, and ξ

**2:** A ← A0

**3:** **for** i = 1 to n **do**

**4:** λi ← 0

**5:** ξi ← u **if** yi = 1 **else** l

**6:** **end for**

**7:** # Optimise A

**8:** **repeat**

**9:** **for** i = 1 to n **do**

**10:** obtain i-th instance (w1, w2, yi) from X and y

**11:** d ← h(w1, w2; A) in (8)

**12:** δ ← 1 if yi = 1 else −1

**13:** α ← min(λi, δ(1/d − γ/ξi)/2)

**14:** ξi ← γξi/(γ + δαξi)

**15:** λi ← λi − α

**16:** β ← δα/(1 − δαd)

**17:** A ← A + βα(w1 − w2)(w1 − w2)⊤A

**18: end for**

**19:** **until** convergence

**20: return** A

Используя полученную метрику расстояния на основе значений h и энкодер, основанный на значении слов в тексте (описанный в первом этапе), вычисляется оценка семантического изменения слов (изменения значений слов в зависимости от контекста):

α(w, C1, C2) = 1/n1n2 · ∑ h(f (w, s1; θ), f (w, s2; θ); A) (3),

s1∈S1(w)

s2∈S2(w)

где C1, C2 – контексты слов (варианты значений), w – целевое слово, s1, s2 – предложения, в которых присутствует целевое слово для соответствующих контекстов, n1, n2 – количества предложений s1, s2. В большинстве подходов к SCD-задаче вычисляется единый общий вектор целевого слова для всего корпуса. В данном подходе вектор слова вычисляется отдельно для каждого предложения, в котором появляется целевое слово. Таким образом вектор слова не смешивает в себе разные значения. Хотя для высокочастотных слов вычисления могут быть довольно затратными, оценка семантического изменения слов может быть эффективно рассчитана, если использовать предвычисленные и проиндексированные эмбединги для всей появлений целевых слов в каждом корпусе предложений.

Для получения эмбедингов, основанных на значениях слов, авторы статьи используют XLM-RoBERTa-large-энкодер. Для обучения метрики расстояния на основе значений используется библиотека ***metric learn*** и алгоритм ITML. Далее, происходит процедура оценки на SCD-задачах. Авторы рассматривают применение подхода для нескольких языков: английский, немецкий, шведский, латинский и русский. В своей работе я ограничился экспериментами только для английского и русского языков.

Для английского языка используется оценочный датасет SemEval-2020 Task1; для русского языка - RuShiftEval. Подробнее данный рассмотрены в части 2.

Авторам удалось добиться результатов, либо сравнимых по качеству с передовыми подходами для данной задачи, либо превосходящих. Наибольшую сложность представляет латинский язык, так как для него нет достаточно большого размеченного датасета для обучения модели. Для остальных языков SDML-подход демонстрирует высокое качество.

* + - 1. **Анализ данных**

Для обучения модели и получения метрики использовался датасет WiC – words in context. Для английского и русского языка два разных датасета, отличающихся по формату. Для английского языка датасет состоит из строковых значений, в которых записаны:

Целевое слово;

Какой частью речи является;

Позиции целевого слова в предложениях;

Пара предложений с целевым словом.

Пример строки датасета:

«carry\tV\t2-1\tYou must carry your camping gear .\tSound carries well over water .\n»

Также каждой строке соответствует метка (T/F), показывающая, является ли значение целевого слова одинаковым для двух предложений. Датасет изначально разделен на тренировочную и тестовую части размерами соотвественно 5428 и 1400 строк-пар предложений.

Для русского языка датасет WiC представляет собой json-файл, то есть набор словарей. Пример словаря:

{'idx': 2,

'word': 'засада',

'sentence1': 'У нас вообще […] засада с героями, способными дотягивать в жизни до собственного творчества',

'sentence2': 'Там в воскресенье все магазины закрыты – вот ведь засада!',

'start1': 17,

'end1': 24,

'start2': 50,

'end2': 57,

'label': True,

'gold\_sense1': 2,

'gold\_sense2': 2}

Как видно, WiC датасет для русского языка также содержит ключевое слово, два предложения, позиции целевого слова в каждом предложении, и бинарную метку True/False, указывающую на совпадение/несовпадение значений слова в предложении соответственно. Размер тренировочной части 19845 пар предложений, тестовой – 8505.

XLM-RoBERTa-large-энкодер на вход принимает пару предложений и позиции целевого слова в предложении. На выходе получаем вектор эмбединга.

Для оценки эффективности метода для английского используется датасет SemEval-2020 Task1. Это два объемных текста (253644 и 353692 предложений) и набор из 37 целевых слов. Задача состоит в том, чтобы для каждого слова определить, совпадают ли его значения в каждом из текстов. Даны также бинарные экспертные оценки (0 – значение не меняется, 1 – значение меняется). Для упрощения также даны готовые лемматизированные варианты текстов.

Для оценки эффективности метода для русского языка используется датасет RuShiftEval. В этом датасете необходимо сравнить значения одного и того же слова при его употреблении в разные исторические промежутки времени. Иными словами, определить, менялось ли значение со временем. Так для каждого целевого слова приведен корпус предложений. Для каждого предложения есть метка, для какого промежутка времени это предложение и, соответственно, значение целевого слова характерно. Например, для слова «авторитет» приведены такие предложения, как: «Я отстаивала свою самостоятельность, он - свой авторитет» и «Полагаясь больше на авторитет власти, чем на свой, она кинулась навстречу городовому; бежала и кричала изо всей силы "караул!"». Для первого предложения указан промежуток времени 1700-1916гг., для второго - 1918-1990гг. Также указаны метки положения целевого слова в каждом предложении. Для оценки приведены средние значения экспертных оценок. Оценки выставлялись людьми в соответствии со шкалой:

0 - Трудно определить

1 - Значения разные

2 - Значения сильно дистанцированы

3 - Значения близки

4 - Значения полностью идентичны

Так, например, для слова «авторитет» средняя экспертная оценка равна 3.23, то есть значение слова в исторических периодах с 1700 по 1916 годы и с 1918 по 1990 годы практически не менялось.

* + - 1. **Методика повторения эксперимента**

**3.1 Обучение моделей**

Для повторения эксперимента, приведенного в статье и проверки результатов были установлены необходимые библиотеки. Для получения эмбедингов в среде разработки Google Colab из репозитория GitHub импортирован и установлен семантический энкодер XL-LEXEME. Энкодер состоит из двух частей. Первая – класс InputExampe – принимает на вход строку и список из двух целочисленных значений – начальный и конечный индексы символов, составляющих целевое слово, и выдает объект, который подходит для преобразования его в вектор эмбединга. Вторая часть – непосредственно сам семантический би-энкодер.

Сначала возьмем английский язык. Обучающий датасет WiC находится в открытом доступе на платформе GitHub. Архив датасета также загружаем в колаб и распаковываем. Пример строки датасета:

«carry\tV\t2-1\tYou must carry your camping gear .\tSound carries well over water .\n»

Таким образом, для получения эмбедингов сначала необходимо провести парсинг датасета – извлечь предложения и позиции целевого слова в них. Эту задачу выполняет функция «paser». Она обрабатывает тренировочную и тестовую части датасета. Выделенные предложения и позиции целевого слова обрабатываются классом InputExampe и полученные пары объединяются в кортеж. Также обрабатываются метки соответствия/несоответствия: строковые значения «F» и «T» заменяются на -1 и 1 соответственно. Метки объединяются в кортеж с кортежем из пар InputExampe. На выходе получаем список кортежей.

Далее необходимо получить непосредственно эмбединги. Для этого написана функция «get\_prepared\_data». Ей на вход подается список, полученный функцией «parser», на выходе получаем два списка: список векторов эмбедингов и список меток. Списки согласованы по индексам.

На следующем шаге устанавливается и импортируется библиотека «metric\_learn», содержащая модель ITML. Обучение модели осуществляется с помощью функции «train\_itml». В ней происходит обучение моделей с разными значениями гиперпараметра регуляризации γ – 1\*10-5, 1\*10-4, 1\*10-3, 1\*10-2, 1\*10-1, 1\*100, 1\*101 – для поиска наиболее эффективной модели. Стоит отметить, что авторы статьи рассматривали варианты параметра γ вплоть до 1\*105, но при увеличении значения γ значительно вырастает время обучения модели. Поэтому было принято решение ограничить максимальное значение γ до 10. Модель ITML принимает на вход набор эмбедингов и метки. Каждая модель сохраняется в формате pickle.

Затем оцениваем качество моделей. Для этого была написана функция «evaluate\_model», которая считает accuracy\_score между предсказанием модели и правильными ответами на тестовой части датасета.

Аналогичные шаги повторяем для WiC-датасета на русском языке. Отличаться будет парсинг, а также набор параметров γ. Дело в том, что русский датасет почти в 4 раза больше и время на обучение моделей значительно выше. Поэтому максимальное значение гиперпараметра γ ограничено 0.1. Оценка качества также аналогична английскому языку.

**3.2 Задача RusShiftEval**

Для задачи RusShiftEval необходимо оценить, как изменилось значение слов в разные исторические периоды. Сначала нужно подготовить данные. Формируем словарь со словарями: ключ словаря – целевое слово, значение – словарь с двумя ключами – метки исторических периодов, значениями которых будут списки уже обработанных предложений с позициями целевых слов. Схематично это выглядит так:

**…**

[InputExample1,… InputExamplem]

[InputExample1,… InputExamplem]

[InputExample1,… InputExamplem]

[InputExample1,… InputExamplem]

Далее проходимся по полученному словарю и создаем два новых словаря, каждый для своего исторического периода. Ключи словарей – целевые слова, значения – векторы-эмбединги, полученные семантическим би-энкодером.

Загружаем наилучшую модель с гиперпараметром γ = 0.1. На ее основе создаем объект класса get\_metric, который будет возвращать значение метрики simantic change distance между двумя входными векторами. Для каждого слова сравниваем попарно все его употребления в разные периоды и выводим среднее значение метрики simantic change distance.

**3.3 Задача SemEval-2020 Task1**

Для задачи SemEval-2020 Task1 загружаем список целевых слов и два текста. Для упрощения используем сразу лемматизированные варианты текстов. Также, как и в предыдущий подразделах, сначала нужно распарсить данные и отфильтровать только предложения, содержащие целевые слова. Эту задачу выполняет функция «parser\_semeval», которая возвращает два словаря: для обоих ключи – целевые слова, для первого значения – списки предложений, содержащих целевые слова, для второго – кортежи из начальных и конечных позиций целевых слов.

Далее получаем эмбединги с помощью функции «encoder\_semeval», которая принимает два словаря с предложениями и два словаря с позициями слов, подготавливает предложения и передает их в энкодер. Выход функции – два словаря с ключами-словами и значенияма-эмбедингами.

Для оценки сначала попробуем сравнить бинарные метки – совпадаю/не совпадают значения – предсказанные моделью с эталонными значениями. Для этого предсказываем метки для всех возможных пар предложений из двух корпусов для каждого слова и выводим среднее значение метки для каждого слова.

Второй вариант оценки – через коэффициент корреляции предсказанной метрики simantic change distance и эталонных метрик схожести/различности значений (как в задаче RusShiftEval).

* + - 1. **Итоги обучения модели и полученные результаты**

После обучения моделей для английского языка и их оценки взглянем, на результаты:

score itml\_gamma=1e-05= 0.5442857142857143

score itml\_gamma=0.0001= 0.5464285714285714

score itml\_gamma=0.001= 0.54

score itml\_gamma=0.01= 0.5407142857142857

score itml\_gamma=0.1= 0.5735714285714286

**score itml\_gamma=1= 0.5935714285714285**

score itml\_gamma=10= 0.575

Видим, что все модели отработали чуть лучше 50%, наилучший результат при γ = 1 составляет 59%. Чтобы понять много это или мало, посмотрим на результате на основе косинусного расстояния. Для косинусного расстояния точность составила 50%, что соответствует простому угадыванию. Таким образом, даже сама худшая ITML-модель показывает лучшие результаты.

Посмотрим на результаты обучения для русского языка:

score itml\_gamma=1e-05= 0.6320987654320988

score itml\_gamma=0.0001= 0.6338624338624339

score itml\_gamma=0.001= 0.6353909465020576

score itml\_gamma=0.01= 0.6378600823045267

**score itml\_gamma=0.1= 0.660082304526749**

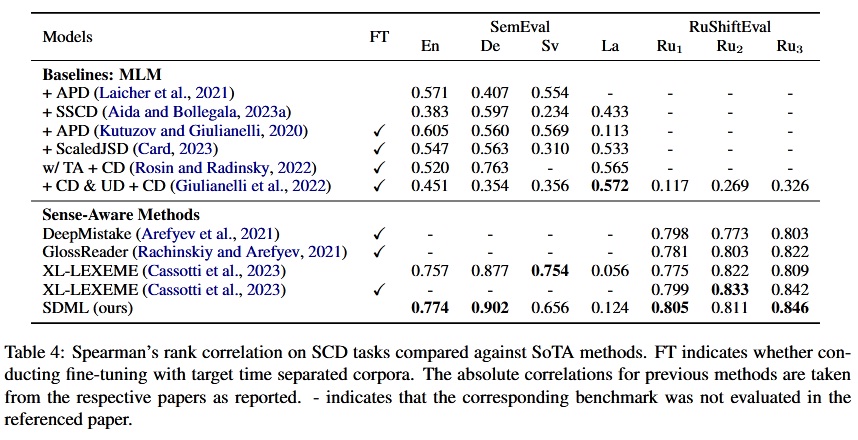
Как видим, наилучший результат достигнут при γ = 0.1 – 66%. Для косинусного расстояния точность очень низкая – менее 37%. Таким образом видно, что для русского языка модель ITML показывает гораздо более высокие результаты. Это можно объяснить тем, что обучающий датасет был гораздо больше и модель обучилась лучше.

Теперь посмотрим, как полученные модели решают задачу RusShiftEval. Поскольку эталонные метки – средние значения экспертных оценок в пределах от 0 до 4, а метрика simantic change distance колеблется в пределах от 1.6432446640999006 до 1.2954883994658588, для их сопоставления используем корреляцию. Она равна -0.43 при pvalue=2.728270337733736e-06, что является статистически-значимой оценкой. Значение отрицательное, так как чем ниже экспертный балл, тем сильнее поменялось значение, но чем выше полученная метрика, тем сильнее отличаются значения. В целом, тенденции совпадают, однако значение коэффициента корреляции довольно низкое.

Для задачи SemEval-2020 Task1 проведем такую же оценку, что и для RusShiftEval, с помощью корреляции, а также сравним бинарные метки. Коэффициент корреляции получился равным 0.28 при pvalue=0.14815160592833443, что говорит о практическом отсутствии статистически-значимых закономерностей. Предсказанные средние значения бинарных меток для всех слов оказались либо равными 1, либо очень близкими к 1. То есть модель практически не видит семантических изменений значений слов между двумя корпусами предложений, что не согласуется с экспертными оценками.

* + - 1. **Выводы и заключение**

В ходе изучения статьи и повторения эксперимента к сожалению не удалось добиться сопоставимых с оригиналом результатов. В таблице 1 приведены результаты авторов статьи на задачах SemEval для английского, немецкого, шведского, латинского языков и RusShiftEval для русского языка. Значения таблице – коэффициент корреляции Спирмена.

Таблица 1. Коэффициенты корреляции Спирмена для разных подходов к решению задачи Semantic Change Distance.

Относительно неплохие результаты показала модель, обученная на русскоязычном датасете на задаче RusShiftEval. Скорее всего это связано с тем, что для обучения модели использовался больший объем данных, чем для англоязычной модели. Тем не менее, коэффициент корреляции почти в 2 раза меньше, чем полученный авторами статьи. С задачей SemEval-2020 Task1 полученная мною модель вообще не справилась.

Стоит отметить, что из-за ограниченных вычислительных мощностей и времени проведения экспериментов не удалось обучить ITML-модели с большими значениями гиперпараметра γ. Поскольку авторы статьи не указывают, для какого значения γ приведены финальные результаты, возможно при больших значениях гиперпараметра происходит резкий скачок в качестве работы модели.

Возможными вариантами развития данной работы являются:

1. Проверка работы метода на других языках. Так, авторы рассматривают, помимо русского и английского, немецкий, шведский и латинский языки.
2. Основная проблема – ограниченный набор данных для обучения и тестирования моделей. Поэтому для улучшения результатов есть смысл собрать больше данных и расширить датасеты. (Однако, такой подход требует существенных вложений и доступен скорее большим технологическим компаниям).
3. Дополнительное исследование влияния гиперпараметра γ.
4. Использование подхода на других схожих задачах и датасетах (например, датасет AM2iCo).
   * + 1. **Список источников**

1. Taichi Aida, Danushka Bollegala. «A Semantic Distance Metric Learning approach for Lexical Semantic Change Detection».

2. Pierluigi Cassotti, Lucia Siciliani, Marco de Gemmis, Giovanni Semeraro and Pierpaolo Basile. «XL-LEXEME: WiC Pretrained Model for Cross-Lingual LEXical sEMantic change»

3. Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. «Information-theoretic metric learning»

4. <https://github.com/pierluigic/xl-lexeme>