МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКО-АРМЯНСКИЙ УНИВЕРСИТЕТ

На правах рукописи

Аветисян Карен Ишханович

Метод обнаружения межъязыковых заимствований в текстах

Специальность 2.3.5 —
«Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
к.ф.-м.н.
Турдаков Денис Юрьевич

Москва — 2023

Оглавление

Введение

В современном мире обнаружение текстовых заимствований является важной задачей в обеспечении честной и справедливой оценки научных работ. Текстовым заимствованием считается процитированный или использованный без должного цитирования фрагмент текста.

С развитием современных систем машинного перевода особую сложность для выявления стали представлять заимствования, совершенные из ресурсов других языков, такие заимствования называются межъязыковыми. Сложность выявления подобного рода заимствований и отсутствие инструментов их обнаружения для многих языков актуализируют данную задачу.

Особенно остро задача стоит для научных работ, написанных на языках, являющихся малоресурсными. *Малоресурсные языки* - это те языки, для которых существует малое количество данных в цифровом виде. Малое количество ресурсов на определенном языке приводит к совершению заимствований из ресурсов других языков.

Существующие методы обнаружения межъязыковых заимствований опираются на использовании инструментов машинного перевода, мультиязычных тезаурусов, векторных представлений слов. Также в некоторых методах используются инструменты разрешения лексической неоднозначности слов, которые являются специфичными для конкретных языков. Недостатками подобных методов являются их применимость к очень ограниченному количеству языков, обычно не являющихся малоресурсными, или, при неимении такого ограничения, низкое качество работы для малоресурсных языков. Таким образом, разработка метода обнаружения межъязыковых заимствований, применимого к большому количеству языков, в том числе малоресурсных, является актуальной проблемой.

Примером малоресурного языка может служить армянский язык. Для армянского языка не существует системы обнаружения межъязыковых

заимствований, что открывает возможности использования подобного типа заимствований и актуализирует задачу разработки подобной системы.

Объектом исследования диссертации являются текстовые документы, написанные на литературном языке, предметом — анализ оригинальности текстовых документов в условиях, когда возможно их полное или частичное заимствование из текстов, написанных на другом языке. Литературный язык - это наднациональный язык, который был приведен к общим письменным нормам для его использования в качестве официального.

Задача ставится следующим образом: имеяя набор с большим количество документов-источников на одном языке, в анализируемом документе на другом языке требуется найти и сопоставить те фрагменты, которые были заимствованы из фрагментов этих документов-источников. *Документы-источники* - это документы, из текстов которых могло быть произведено заимствование, *анализируемые (подозрительные) документы* - это документы, в которых потенциально возможно содержание межъязыковых заимствований.

Целью работы является разработка метода и программных средств обнаружения заимствований между текстами различных языков, в том числе применимого к малоресурсным языкам.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Исследовать существующие методы обнаружения межъязыковых текстовых заимствований.
2. Разработать и реализовать метод обнаружения межъязыковых заимствований между текстами различных языков, также применимый к текстам малоресурных языков.
3. Провести экспериментальное сравнение существующих методов с разработанным методом с использованием общепринятых метрик качества и эталонных наборов тестовых данных.
4. Проверить алгоритм на уязвимость к искусственным атакам.

5. На основе разработанного метода создать программные средства для обнаружения межъязыковых заимствований.

Научная новизна: Разработан новый метод обнаружения межъязыковых заимствований, с использованием собранного в рамках диссертационной работы словаря “межъязыковых синонимов”. Разработанный метод применим к малоресурсным языкам и, в отличие от других подобных методов, не использует инструменты машинного перевода и разрешения лексической многозначности слов. Метод показывает более высокое качество обнаружения межъязыковых заимствований, по сравнению с методами, результаты которых представлены в открытом доступе [1].

Практическая значимость заключается в разработке программных средств обнаружения межъязыковых заимствований, которые возможно использовать в работе высших учебных заведений, в том числе тех стран, в которых государственный язык является малоресурсным [1-6]. Метод, в частности, опробован на текстах армянского языка, и, исходя из полученных результатов, применим в работе с текстами армянского языка. Метод был дополнен детальным анализатором и алгоритмом склейки обнаруженных текстовых фрагментов, которые используются в программной системе “Антиплагиат.ру”. Дополнение изначального метода дало дополнительный прирост в эффективности обнаружения заимствований. Таким образом, дополненный метод может быть использован в качестве нового метода обнаружения межъязыковых заимствований с лучшей эффективностью.

Представлена методика выбора языковой модели для этапа детального анализа, учитывающая угрозу возможности осуществления искусственных атак.

Также, были сгенерированы тестовые выборки обнаружения межъязыковых заимствований в двух различных настройках: где в рамках одного анализируемого документа возможно содержание заимствований из текстов на нескольких различных языках, и, где в рамках одного анализируемого документа возможно содержание заимствований из текстов только на одном языке [1].

Основные положения выносимые на защиту:

1. Разработан новый метод обнаружения межъязыковых заимствований, превосходящий по эффективности существующие. Дополнительно, метод применим к задаче обнаружения межъязыковых заимствований в текстах малоресурсных языков.
2. Разработан новый метод генерации словаря “межъязыковых” синонимов, позволяющего достичь высоких показателей метрики полноты для этапа извлечения кандидатов в задаче обнаружения межъязыковых текстовых заимствований.
3. Разработан новый метод генерации искусственных атак “черного ящика” на языковые модели бинарной классификации, превосходящий по доле успешных атак, а также по дистанции Левенштейна и семантической близости все существующие аналоги.
4. Разработана методика выбора языковой модели для этапа детального анализа, учитывающая угрозу возможности осуществления искусственных атак.

Апробация работы. Результаты данной работы докладывались на конференциях, форумах:

1. XIV Годичная научная конференция РАУ, 2021, Ереван, РА;
2. LREC 2022 Workshop on Processing Language Variation: Digital Armenian (DigitAm), 2022, Марсель, ФР;
3. Международная конференция ”Иванниковские чтения 2022”, Казань, РФ;
4. AINL: Artificial Intelligence and Natural Language Conference, 2023, Ереван, РА;
5. DataFest Yerevan, 2023, Ереван, РА.

Публикации. По теме диссертации опубликовано 4 печатных работ, в том числе в изданиях и сборниках научных конференций индексируемых в Scopus [3-5], а также 1 свидетельство о государственной регистрации программы для ЭВМ [2].

Личный вклад. Предлагаемые в диссертации инструменты, текстовые наборы данных и исследования разработаны и выполнены автором или при его непосредственном участии.

Внедрение результатов. Результаты, полученные в рамках данной работы, внедрены в инструмент обнаружения заимствований “Sieve”, который в свою очередь внедрен в следующих учреждениях:

1. Российско-Армянский Университет
2. Высший Аттестационный Комитет Республики Армения;

Объем и структура работы. Диссертация состоит из введения, четырёх глав, заключения и двух приложений. Полный объем диссертации составляет 139 страницы текста, включая 21 рисунок и 39 таблиц. Список литературы содержит 111 наименований.

Глава 1. Обзор литературы

Большое количество исследований было посвящено решению задачи обнаружения межъязыковых заимствований [7-9]. Множество подходов к решению данной задачи опираются на устранении разницы между языками. Самым очевидным решением устранения разницы между языками является использование машинного перевода для приведения всех используемых в рамках сравнения текстов к одному языку с последующим использованием моноязычных методов поиска заимствований [10-14]. Однако, подобные методы сильно зависимы от качества и доступности используемых инструментов машинного перевода. Для некоторых языков подобные инструменты или низкого качества или вовсе отсутствуют. Рассматривая, например, систему машинного перевода Google Translate, которая является одной из самых популярных подобных систем на момент 2023 года с более чем ста миллионами пользователей[1] и поддерживает 133 языка, в [15] показано, что для многих языков данная система имеет низкое качество. Дополнительно, процесс машинного перевода является времязатратным процессом, что также влияет на использование подобных методов. Некоторые методы основываются на использовании различных параллельных наборов данных. Используя параллельные тексты на разных языках, подобные методы пытаются обучиться проецированию векторных представлений документов различных языков в одну гиперплоскость с последующим поиском ближайших [16, 17]. С использованием подобных методов также возникают проблемы их применимости к множеству языков из-за недоступности подобных параллельных наборов данных для этих языков. Также, активно используются методы основанные на тезаурусах с использованием, например, мультиязычных семантических связей между словами. Тезаурус[2] – словарь, собрание сведений, корпус или свод, полномерно охватывающие понятия, определения и термины

---

[1] https://locize.com/blog/google-translate-accuracy/#:~:text=Google%20Translate%20is%20one%20of,language%20experts%20and%20casual%20users.

[2] https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%B7%D0%B0%D1%83%D1%80%D1%83%D1%81

специальной области знаний или сферы деятельности; в современной лингвистике — особая разновидность словарей, в которых указаны семантические отношения (синонимы, антонимы, паронимы, гипонимы, гиперонимы и т. п.) между лексическими единицами. В рамках данных подходов проблемой является лексическая неоднозначность слов, решения которой улучшает результаты поиска [18]. Однако, решение лексической неоднозначности слов является сложной задачей, в частности для лексически богатых и малоресурсных языков. Большинство современных подходов к решению лексической неоднозначности слов основываются на применении BERT-основанных моделей [19-21]. BERT (Bidirectional Encoder Representations from Transformers) [22] – языковая модель, основанная на архитектуре трансформер [23], предназначенная для предобучения языковых представлений с целью их последующего применения в широком спектре задач обработки естественного языка. Однако такого рода модели являются времязатратными из-за большого количества параметров содержащихся в них.

В большинстве существующих алгоритмов обнаружения межъязыковых заимствований используется двухэтапный подход [24]. Первый этап извлечения кандидатов нацелен на быстродействующее уменьшение количества документов-кандидатов для конкретного анализируемого документа в процессе поиска. На втором этапе детального анализа документы-кандидаты извлеченные на первом этапе проходят более детальную проверку для нахождения конкретных фрагментов текстов из которых было произведено заимствование. В следующих двух разделах рассмотрим различные существующие решения каждого из этих этапов по отдельности.

1.1. Извлечение кандидатов

В методе представляемом в [25] используется метод основанный на векторном представлении n-граммов букв и применим для нахождения заимствований между синтаксически и лексически близкими языками.

Некоторые алгоритмы основываются на извлечении и использовании различного рода информации из больших мультиязычных корпусов. Подобный подход используется в методе “Cross-Language Explicit Semantic Analysis” (CL-ESA) [16], который является мультиязычной версией алгоритма “Explicit Semantic Analysis” (ESA) [26]. Имея коллекцию документов D для каждого документа d алгоритм ESA строит его векторное представление, которое основано на TF-IDF близости между рассматриваемым документов и различных меток концептов встречающихся в статьях Wikipedia[3]. Алгоритм CL-ESA работает тем же образом только вместо использования моноязычной коллекции, используется мультиязычная коллекция где метки концептов являются общими для статей различных языков. Другой метод основывающийся на применении мультиязычных корпусов “Cross-Language Alignment-based Similarity Analysis” (CL-ASA) [17] использует двуязычный словарь полученный с помощью “ IBM alignment model 1” [27-29], где для каждого слова есть его перевод и вероятность подобного перевода. Тем самым алгоритм подсчитывает возможную вероятность одного фрагмента быть переводом другого.

Большое количество алгоритмов основываются на использовании машинного перевода для приведения анализируемого текста к языку проверочной коллекции и последующем применении алгоритмов моноязычного поиска [10-14]. Методы использующие мультиязычные тезаурусы основываются на приведении текстов в независимую от языка форму с их последующим сравнением. Такие тезаурусы как BabelNet [30], OpenThesaurus[4], и EuroWordNet [31] являются мультиязычными семантическими сетями включающими в себя различные связи

[3] https://www.wikipedia.org/
[4] https://www.openthesaurus.de/

между текстовыми единицами. Так, например, с использованием EuroWordNet алгоритм “MLPlag” [32] извлекает независимые от языков смысловые концепции слов и производит сравнение между данными концепциями. Метод “Cross-Language Conceptual Thesaurus based Similarity” (CL-CTS) [33] с использование тезауруса смысловых концептов Eurovoc[5] представляет документы в виде векторов, дальнейшее сравнение производится с помощью подсчета косинусной близости между данными векторами. Другой метод “Cross-Language Knowledge Graphs Analysis” (CL-KGA) [34, 35] основывается на построении графов знаний с использованием тезауруса BabelNet. Подобные графы знаний конструируются для каждого из документов с использованием смысловых концептов слов и связей между ними. Схожесть между документами считается на основе алгоритма близости графов. Данный метод основывается на решении задачи лексической неоднозначности слов, что делает его неприменимым для многих малоресурсных языков.

В [36] представляется метод объединяющий в себе разные подходы описанные выше, что приводит к улучшению результатов поиска. Данный метод по отдельности использует 2 метода основанных на переводе и моноязычном поиске и на использовании модифицированного алгоритма CL-ESA. После чего, получая оценки для топ 10 ближайших кандидатов от каждого метода считает усредненную оценку и возвращает документы исходя из данной усредненной оценки.

В рамках данной работы используется метод основанный на тезаурусе, который обходит задачу определения лексической неоднозначности слов на этапе построения словаря приводящего тексты в независимую от языка форму.

[5] https://op.europa.eu/s/vFSH

1.2. Детальный анализ

Методы “CL-CTS-WE” и “CL-WES” [37] решают проблему обнаружения конкретных фрагментов из которых было произведено межъязыковое заимствование с помощью векторного представления слов (вещественных векторов в пространстве с фиксированной невысокой размерностью). Первый метод использует топ 10 ближайших по векторной близости (подсчет векторной близости осуществляется с помощью косинусного сходства) слов к рассматриваемому слову, с помощью данных слов составляется мешок слов. Второй метод представляет фрагменты текста в качестве суммы векторов его слов. Мультиязычность данных векторов обеспечивает инструмент MultiVec [38]. Другой метод использующий векторные представления слов [39] основывается на мультиязычных векторах слов представленных в библиотеке “MUSE: Multilingual Unsupervised and Supervised Embeddings”[6] [40, 41] компанией Facebook и сравнении графов на уровне предложений. Векторы слов являются мультиязычными, если для слов различных языков их вектора приведены в одно многомерное пространство. Графы для предложений-источников строятся по n-граммам слов, после чего при проходе по n-граммам слов анализируемых предложений строится граф сопоставимости двух предложений. Близость определяется сравнением графа сопоставимости с графом предложения-источника.

В представляемом в [42] методе решается проблема детального анализа путем попарного сравнения предложений анализируемого документа с предложениями документов-кандидатов. Попарное сравнение приводится к задаче бинарной классификации - является ли одно предложение переводом другого. Классификация производится с использованием BERT-основанных моделей.

В рамках представляемого метода на этапе детального анализа был использован подход представленный в [42] с использованием мультиязычной

[6] https://github.com/facebookresearch/MUSE

языковой модели XLM-RoBERTa [43], которая является межъязыковым текстовым энкодером и была обучена на 2.5 терабайтах данных для 100 языков.

1.3. Выводы

В рамках данной главы было дано общее представление двухэтапной структуры большинства алгоритмов нахождения межъязыковых текстовых заимствований.

Для этапа извлечения кандидатов были рассмотрены различные методы основанные на переводе и моноязычном поиске, на использовании параллельных больших текстовых корпусов, мультиязычных тезаурусов, а также основанные на объединении данных методов. Проанализировав слабые стороны различных алгоритмов, в рамках данной работы этап извлечения кандидатов был реализован с использованием мультиязычного тезауруса при этом обходя решение проблемы лексической неоднозначности слов.

Для этапа детального анализа были рассмотрены методы основанные на использовании мультиязычных векторов слов, подсчете близости между графами фрагментов текстов, а также основанные на использовании языковых моделей. В рамках данной работы для обеспечения максимальной мультиязычности представляется метод основанный на использовании языковой модели поддерживающей 100 языков.

Глава 2. Предлагаемый метод

В данной главе описывается представляемый в рамках работы метод обнаружения межъязыковых заимствований [1]. Представляемый метод разбит на 2 этапа: извлечение кандидатов и детальный анализ. В первом разделе представляется этап извлечения кандидатов, который основывается на использовании мультиязычного тезауруса и обходит проблему определения лексической неоднозначности слов. Во втором разделе представляется этап детального анализа, основывающийся на бинарной классификации - является ли одно предложение переводом другого.

2.1. Извлечение кандидатов

В этом разделе описан процесс первичной фильтрации кандидатов из проверочной базы, позволяющий резко снизить количество документов или текстовых фрагментов подлежащих более дорогостоящему процессу детального анализа. В качестве проверочной базы могут служить как заранее собранные коллекции документов, так и тексты из Интернета, при этом тексты проверочной базы написаны на отличном от анализируемого документа языке.

В первом подразделе описывается процесс фильтрации кандидатов на уровне фрагментов текстов документов. Во втором подразделе представлены методы сравнения близости текстов используемые в рамках процесса извлечения кандидатов. Третий подраздел посвящен представлению слов в независимой от языка форме с использованием векторных представлений слов, а также многоязычных тезаурусов. В четвертом подразделе описаны эксперименты проведенные для оценки работы этапа извлечения кандидатов. Пятый подраздел посвящен методам предобработки документов, а также предварительной фильтрации кандидатов на уровне сокращения изначальной проверочной базы

путем удаления документов-источников написанных на диалектах используемых языков.

2.1.1. Процесс извлечения текстовых фрагментов-кандидатов

В рамках представляемого метода поиск релевантных кандидатов производится на уровне фрагментов текстов документов. Тексты анализируемых документов, а также документов-источников разбиваются на более мелкие фрагменты, такие как: предложения и параграфы соответственно.

Таким образом, наша задача на данном этапе состоит в том, что имея коллекцию из *N* документов, разбитую на N_m фрагментов на некотором языке L_1, в качестве коллекции источников, и некоторый фрагмент S_i из анализируемого документа S на языке L_2 $(L_1 \neq L_2)$, для фрагмента S_i отфильтровать топ k релевантных ему фрагментов из имеющихся N_m, где $k << N_m$.

В качестве метода фильтрации релевантных фрагментов используется метод, основанный на построении инвертированного индекса. Инвертированный индекс - это структура данных, используемая для индексации баз данных для последующего быстрого поиска в них. Основная идея инвертированного индекса состоит в том, что при наличии некоторых множеств с их элементами, каждому элементу сопоставляются те множества, в которых он встречается. В нашем случае в качестве множеств служат фрагменты-источники, а в качестве элементов - слова данных фрагментов. Таким образом, для каждого слова в запросе заранее известно в каких фрагментах оно встречалось. При использовании множества слов в запросе, возвращается пересечение множеств фрагментов найденных для каждого из этих слов.

В нашем случае, инвертированный индекс строится по прошедшим предобратку фрагментам-источникам. Предобработка фрагментов проходит в несколько этапов. Для начала производится токенизация фрагментов, затем их

лемматизация, все стоп-слова и пунктуационные символы удаляются, а также все заглавные буквы приводятся к строчным. Последним этапом предобработки является представление слов фрагментов в независимой от языка форме, данный процесс представлен в 2.1.3. В качестве фрагментов для документов-источников используются параграфы.

В свою очередь, анализируемые документы разбиваются на анализируемые предложения, после чего данные предложения подвергаются такому же процессу предобработки.

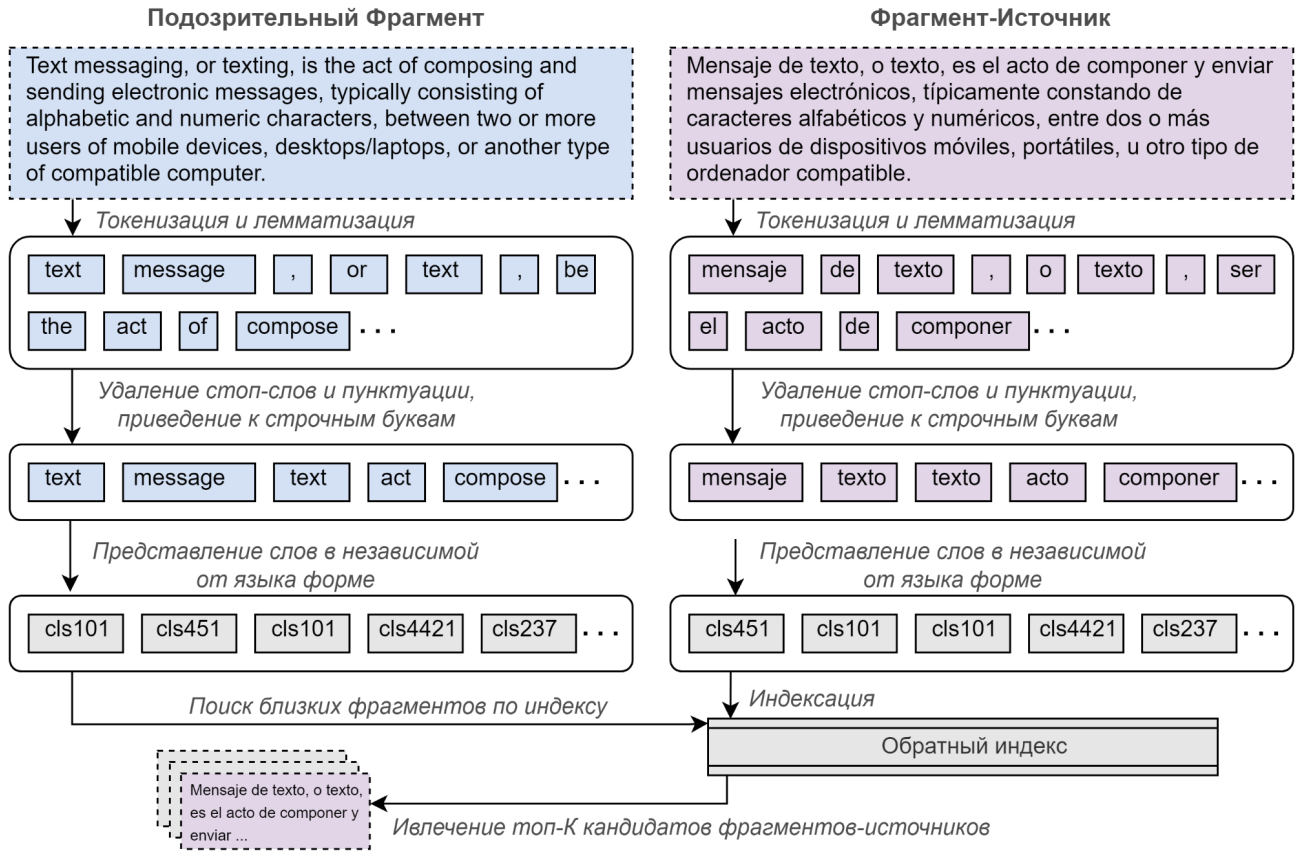

Рисунок 2.1 – Процесс предобработки анализируемых фрагментов и фрагментов-источников, приведение их к независимой от языка форме и процесс поиска фрагментов-кандидатов.

Таким образом, поиск релевантных параграфов-источников производится на уровне анализируемых предложений методом полнотекстового поиска с использованием инвертированной индексации и функции оценки близости между

двумя текстовыми фрагментами, для быстрого поиска, и в независимой от языка форме. Данный процесс описан на рисунке 2.1.

2.1.2. Методы сравнения близости текстов

2.1.2.1. Коэффициент Жаккара

Коэффициент Жаккара оценивает близость между двумя конечными множествами (в нашем случае множествами слов), и определяется как отношение мощности пересечения двух рассматриваемых множеств A и B к мощности их объединения:

$$J(A,\ B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (2.1)$$

Область определения функции коэффициента Жаккара равна [0; 1], где функция равна 0 при полном отличии элементов двух проверяемых множеств, и равна 1 при их полном равенстве.

Недостатком использования данного метода является его зависимость от длин множеств. Таким образом, в случае когда множество A является подмножеством множества B ($A \subset B$), и мощность множества B сильно больше мощности множества A ($|B| >> |A|$), значение коэффициента Жаккара будет стремится к 0.

2.1.2.2. Мера Шимкевича-Симпсона

Мера Шимкевича-Симпсона, также оценивает близость между двумя конечными множествами A и B (в нашем случае множествами слов). Данная мера

определяется отношением мощности пересечения двух множеств к меньшей из двух мощностей отдельных множеств:

$$overlap(X, Y) = \frac{|X \cap Y|}{min(|X|, |Y|)} \quad (2.2)$$

Мера Шимкевича-Симпсона определена на отрезке [0; 1]: равна 0 при полном отличии элементов двух проверяемых множеств, и равна 1, если одно из множеств является подмножеством другого.

В данной мере учитывается недостаток коэффициента Жаккара путем деления на меньшую из двух мощностей рассматриваемых множеств, вместо их объединения. Таким образом, в ситуации, когда множество A является подмножеством B, значение метрики будет равно 1.

2.1.2.3. MinHash

MinHash является вероятностной оценкой близости двух множеств и модификацией коэффициента Жаккара. В отличии от коэффициента Жаккара, метод MinHash не считает объединение или пересечение множеств, вместо этого он основан на использовании хэш-функций.

Предположим, что мы имеем два фрагмента текста, разбитых на множества слов A и B, и хэш-функцию H, считающую хэши для каждого из слов рассматриваемых множеств. Дополнительно, определим функцию H_{min}, которая для каждого элемента множества считает его хэш, а затем возвращает хэш с минимальным значением.

После вычисления $H_{min}(A)$ и $H_{min}(B)$, вероятность того, что $H_{min}(A) = H_{min}(B)$ равна вероятности того, что из всех элементов двух множеств ($A \cup B$), элемент с минимальным значением хэша принадлежит множеству пересечения

рассматриваемых множеств ($A \cap B$). Таким образом, вероятность равенства $H_{min}(A)$ и $H_{min}(B)$ равна коэффициенту Жаккара:

$$P\left(H_{min}(A) = H_{min}(B)\right) = J(A, B) \tag{2.3}$$

Посчитав значения функции H_{min} для двух множеств и сравнив их, мы получим бинарный ответ, который не говорит ничего о степени схожести двух множеств. Для решения данной проблемы вместо одной хэш-функции используются k функций. Число k выбирается исходя из допустимой, для конкретной задачи, величины ошибки ε и равно:

$$k = \frac{1}{\varepsilon^2} \tag{2.4}$$

Отсюда получается, что для оценки близости двух множеств с ошибкой 0,05 требуется 400 различных хэш функций.

2.1.2.4. Okapi BM25

Okapi BM25 [44] является функцией ранжирования, используемая в различных поисковых системах, для нахождения релевантных к запросу документов и их сортировки. BM25 – это одна из функций семейства TF-IDF функций.

Для запроса Q функция BM25 дает оценку релевантности некоторого документа D_k из общей коллекции документов N ($[D_1 \dots D_m] \in N$). Прежде чем перейти к самой функции, определим две ее составляющие - частоту слова (Term Frequency - TF) и обратную документную частоту (Inverse Document Frequency - IDF).

Имея множество слов $(w_1 \dots w_n) \in Q$, частота слова w_i в документе D_k равна:

$$TF(w_i, D_k) = \frac{f_{D_k}(w_i)}{|D_k|}, \quad (2.5)$$

где $f_{D_k}(w_i)$ - сколько раз слово w_i встречалось в документе D_k, а $|D_k|$ - общее количество слов в документе.

Обратная документная частота некоторого слова w_i будет равна:

$$IDF(w_i) = log\frac{|N|}{n(w_i)}, \quad (2.6)$$

где $n(w_i)$ – это количество документов содержащих слово w_i , а $|N|$ - количество документов в коллекции.

Используя вышеописанные функции, сама функция ранжирования BM25 имеет следующий вид:

$$score(D, Q) = \sum_{i=1}^{n} IDF(q_i)\frac{f(q_i, D)(k_1 + 1)}{f(q_i, D) + k_1(1 - b + b\frac{|D|}{avgdl})} \quad (2.7)$$

где $avg(N)$ - средняя длина документов в коллекции, k_1 и b являются свободными коэффициентами. k_1 контролирует влияние одного слова из запроса на значение, получаемое для конкретного документа. b контролирует влияние длины документов на итоговое значение функции.

2.1.3. Представление слов в независимой от языка форме.

Для осуществления этапа фильтрации релевантных параграфов-источников на языке L_1 для некоторого анализируемого предложения на языке L_2, требуется привести их в одну плоскость, т.е. либо привести их к одному языку, либо привести их к независимой от языка форме. В рамках данной работы мы рассмотрим метод приведения фрагментов к независимой от языка форме.

Метод основывается на использовании словаря “межъязыковых” синонимов. Словарь “межъязыковых” синонимов - это словарь, в котором содержатся слова разбитые на определенные нумерованные группы. Группы должны содержать в себе слова на различных языках, при условии, что слова в рамках одной группы, независимо от языка, являются семантически близкими друг к другу:

Группа 995*: сладость, dulce, bonbon, конфета, friandise, confection,* ֆաղցր*, sweet, konfekt,* իրուշակեղեն*, сладкое, лакомство, …*

Таким образом, при наличии подобного словаря, слова в анализируемых фрагментах и фрагментах -источниках могут быть заменены на номер их группы и перейти в независимую от языка форму.

Далее представлены два метода получения подобного словаря “межъязыковых” синонимов, с использованием межъязыковых векторов слов и графа знаний Universal WordNet [45-47] (в дальнейшем: UWN).

2.1.3.1. Межъязыковое векторное представление слов

Учитывая, что наша задача состоит в разбиении слов на группы для словаря “межъязыковых” синонимов, а в данном словаре все слова содержатся в своей лемматизированной форме, следовательно разбиение по группам должно проходить с использованием векторных представлений лемм.

Для русского языка существуют заранее обученные векторы лемм слов, находящиеся в открытом доступе. В нашем случае, были использованы векторы обученные на текстовом корпусе Тайга[7] [48]. Корпус содержит пять миллиардов слов. Векторы были обучены с использованием алгоритма fastText[8] [49,50], с размерностью 300. Итоговое количество векторов равно 192,5 тысячам.

Для армянского языка не существует заранее обученных векторов на уровне лемм, следовательно стояла задача их обучить. В качестве алгоритмов для обучения векторов использовались два алгоритма:

1. Модифицированный с учетом подстрок fastText [4]
2. GloVe[9] [51]

Векторы были обучены на предварительно лемматизированных данных представленных в [52]. Размерность векторов также, как и в случае с леммами русского языка, равнялась 300. В итоге были получены векторы для 110 тысяч лемм армянского языка.

Алгоритм отображения векторов слов в одной гиперплоскости. При наличии моноязычных векторов на двух различных языках, в качестве метода их отображения был выбран метод представленный в статье [40]. Отображение векторов на одной гиперплоскости производится с помощью алгоритма “MUSE: Multilingual Unsupervised and Supervised Embeddings” созданного компанией Facebook. Очевидным условием также является одинаковая размерность векторов двух различных языков.

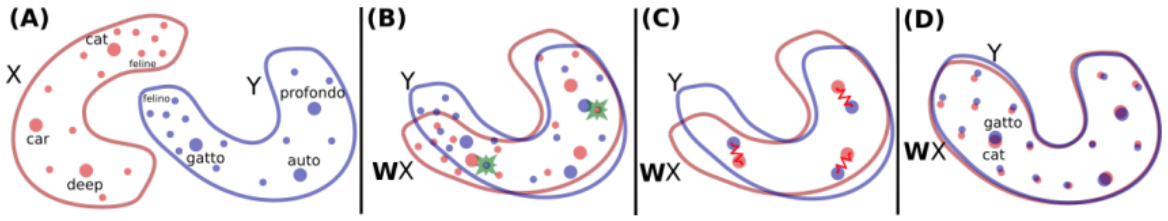

Рисунок 2.2 – Пример проецирования векторов английского и итальянского языков на одну гиперплоскость [40].

[7] https://tatianashavrina.github.io/taiga_site/
[8] https://fasttext.cc/
[9] https://nlp.stanford.edu/projects/glove/

Данный алгоритм основан на использовании такой матрицы переходов, что векторы слов одного языка отображаются на гиперплоскости векторов другого языка таким образом, что вектора семантически близких слов в двух различных языках оказываются близки друг к другу (рисунок 2.2). Нахождение такой матрицы переходов осуществляется с использованием состязательного подхода. В рамках данного подхода имеются два множества векторов на двух различных языках $X_{lang\_1} = \{x_1, \dots, x_n\}$ и $Y_{lang\_2} = \{y_1, \dots, y_m\}$ и матрица переходов W. Существует модель-дискриминатор, целью которой является различение элементов $WX_{lang\_1} = \{Wx_1, \dots, Wx_n\}$ от элементов Y_{lang_2}. При этом матрица W обучается таким образом, чтобы модель-дискриминатор не могла отличать элементы WX_{lang_1} от элементов Y_{lang_2}.

Данный подход был выбран исходя из результатов показанных им для различных пар языков, где для некоторых языковых пар в 80% случаев ближайшими к векторам слов оказывались векторы их переводов.

Кластеризация межъязыковых векторов слов. После проецирования векторов армянского и русского языка в одну гиперплоскость, следующим этапом по получению словаря “межъязыковых” синонимов является этап кластеризации полученных векторов. Изначально для проверки качества векторов лемм слов, процесс кластеризации был произведен отдельно для векторов лемм армянского и русского языков.

В качестве алгоритмов кластеризации были выбраны два алгоритма:

1. K-means
2. K-means-constrained

Алгоритм K-means - это алгоритм кластеризации, основная идея которого заключается в минимизации среднеквадратичного отклонения векторов кластера от его центра. Т. е. функция минимизации будет выглядеть следующим образом:

$$arg\,min_{s} \sum_{i=1}^{k} \sum_{x \in S_i} \left(x - \mu_i\right)^2, \tag{2.8}$$

где k - заранее заданное число кластеров, S_i - i-тый кластер, x - вектор из кластера S_i, μ_i - центр кластера S_i.

Данный алгоритм имеет два недостатка. Первый недостаток заключается в том, что для его использования требуется заранее задать количество кластеров на которые будут разбиты векторы. Вторым недостатком является то, что в один и тот же кластер может попасть большое количество векторов, что для нашей задачи является неверным, т.к. количество семантически близких слов, которые мы хотим видеть в рамках одного кластера, достаточно ограниченно.

Алгоритм K-means-constrained является модификацией алгоритма K-means, в котором заранее задается минимальное и максимальное количество векторов, которые могут принадлежать к одному кластеру.

Получение кластеров моноязычных векторов лемм было произведено тремя разными способами.

K-means. При использовании алгоритма K-means, для кластеризации векторов лемм армянского и русского языков по отдельности, были выбраны следующие параметры количества кластеров: 10000, 8000, 6000. После процесса кластеризации возник вопрос о том, какое максимальное количество слов в рамках одного кластера считать валидным.

Для оценки данного числа, для каждой леммы брались синонимы из интернет-ресурсов “bararanonline.com”[10], для лемм армянского языка, и “synonyms.su”[11], для лемм русского языка. Таким образом, те леммы, для которых на данных интернет-ресурсах содержались синонимы, можно считать “важными” леммами, которые часто встречаются.

[10] https://bararanonline.com/
[11] https://synonyms.su/

Используя эти “важные” леммы, верхний порог количества элементов содержащихся в одном кластере был выбран таким образом, что при отбрасывании кластеров не подходящих под данный порог, в оставшихся кластерах должны были покрываться 90% этих лемм.

Для армянского языка максимальным значением величины кластера стало 25 слов в одном кластере, для русского языка это число достигло 111 слов в рамках одного кластера.

Те кластеры, в которых после процесса кластеризации содержалось большее количество слов, чем в заданных порогах, были отброшены, а леммы, содержащиеся в них, прошли повторную кластеризацию. Процесс продолжался до тех пор, пока во всех кластерах не содержалось лемм меньшему или равному по количеству, чем заданный порог. При этом, при повторной кластеризации лемм, кластеры которых не подпали под порог, количество кластеров выставлялось пропорционально их количеству к общему количеству лемм. Недостатком такого подхода является количество его запусков, для достижения нужного количества слов в каждом кластере.

K-means-constrained. При использовании алгоритма K-means-constrained, верхние пороги для количества элементов в каждом кластере были выставлены в соответствии с числами полученными при кластеризации с помощью обычного K-means. Ключевым недостатком данного метода стала его высокая требовательность к вычислительным ресурсам, для избавления от которой и возникла идея третьего метода кластеризации.

K-means-constrained + разбиение по частям речи. Во избежании проблем с вычислительной мощностью и предполагая, что синонимичные друг к другу слова принадлежат одной и той же части речи, был испробован метод с использованием алгоритма K-means-constrained, на векторах, заранее разбитых по частям речи лемм. Перед передачей векторов слов в алгоритм K-means-constrained, данные слова, в своем текстовом представлении, были переданы совместному синтаксическому парсеру UDpipe[12] [53, 54]. С использованием функции

[12] https://lindat.mff.cuni.cz/services/udpipe/

определения частей речи данного парсера, для представленных слов производилось определение их части речи и дальнейшее группировка по ней. После получения групп слов, для каждой группы по отдельности была произведена кластеризация.

После попыток получения кластеров, по отдельности для лемм армянского и русского языков, тремя описанными методами, уже при визуальном осмотре полученных кластеров стало понятно, что в большом количестве кластеров были объединены слова семантически не являющимися близкими друг к другу. Таким образом, подход к получению словаря “межъязыковых” синонимов с использованием метода кластеризации векторов слов был отброшен.

2.1.3.2. Построение словаря на основании тезауруса

Для решения поставленной задачи возможно использование тезаурусов, хранящих в себе связи между словами на различных языках. С использованием данных связей слова разных языков объединяются в семантически близкие группы. К тезаурусам, хранящим подобные связи, относятся BabelNet [30], DBNary [55], а также различные мультиязычные тезаурусы созданные на основе Princeton WordNet 3.1 [56-58]. Основываясь на количестве слов и на количестве языков, которые покрывает Universal WordNet (UWN), в качестве тезауруса для объединения слов в семантически близкие группы был использован именно он.

2.1.3.2.1. Построение словаря с помощью Universal WordNet.

Universal WordNet (UWN) [45-47] - это большая лингвистическая база, нацеленная на описание слов, сущностей и концептов для более чем 200 языков. UWN содержит более чем полтора миллиона слов на 200 языках, и предоставляет значения этих слов, а также их связь между собой рисунок 2.3.

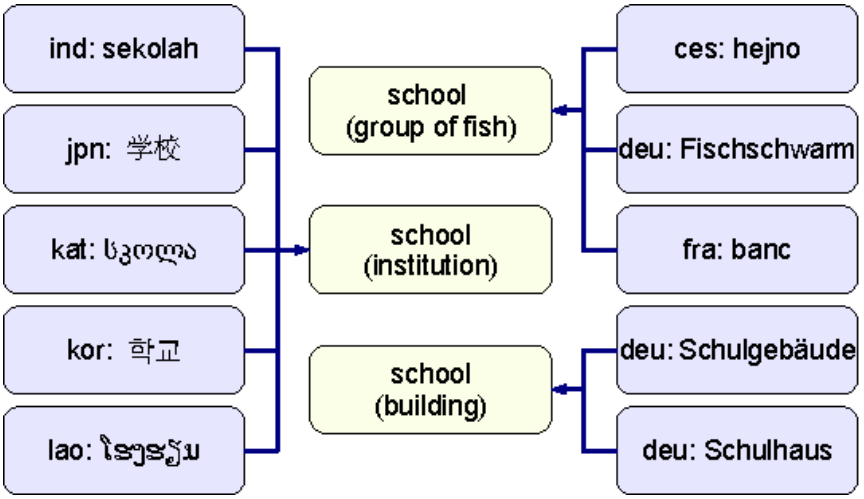

Рисунок 2.3 – Связь различных смыслов одного слова со словами, имеющими тот же смысл на других языках.

В данном тезаурусе, в рамках одного языка, слова объединены в синонимические группы, где каждой такой группе соответствует некая смысловая концепция. Смысловые концепции в свою очередь являются межъязыковыми. Таким образом, каждой смысловой концепции соответствуют синонимические группы множества языков.

Таблица 1 – Количество содержащихся в UWN слов и групп синонимов для каждого рассматриваемого языка

	Группы Синонимов	**Количество Слов**
Армянский	9150	12120
Английский	85937	175208
Русский	31474	50158
Французский	42215	71764
Немецкий	49153	86143
Испанский	36809	59387

Исходя из целей, которые преследовала наша работа, и исходя из того, что UWN предоставляет информацию о связях слов, как межъязыковых, так и внутриязыковых, представлена статистика показывающая количество слов на конкретном из шести рассматриваемых языков (Таблица 1), а также количество внутриязыковых групп (кластеров) синонимов для каждого из этих языков.

Объединяя синонимические группы смысловых концепций для нужных нам языков, получаются группы “межъязыковых” синонимов, которые и образовали базовый вариант нашего словаря. Однако, главной проблемой разбиения слов по смысловой концепции является их смысловая многозначность. Также, одно слово может принадлежать множеству смысловых концепций, что приведет к ситуации, в которой данное слово содержится во множестве групп “межъязыковых” синонимов, что может негативным образом сказываться на процессе извлечения фрагментов-кандидатов. Во избежании данных проблем, словарь был модифицирован путем использования только самых популярных смысловых концепций конкретных слов. Оценка популярности каждого из смысловых концептов определенного слова была извлечена из тезауруса Princeton WordNet 3.1, так как данный тезаурус, для английских слов, содержит в себе те же концепции, что представлены в UWN. Оценка смысловых концептов извлекалась только для слов английского языка, так как Princeton WordNet 3.1 является тезаурусом английского языка. Таким образом, используя самые популярные значения слов, было снижено количество “межъязыковых” синонимических групп, в которых встречается определенное слово.

Пусть L_{UWN} будет множеством языков поддерживаемых в UWN, и пусть C_{UWN} множество всех смысловых концептов во всех языках, и w некоторое слово английского языка. В качестве $S^{l}_{C_w}$ обозначим множество синонимических групп принадлежащих всем смысловым концептам слова w $(C_w \in C_{UWN})$ на языке $l \in L_{UWN}$. Используя данные обозначения, определим базовый и

модифицированный процесс получения “межъязыковых” синонимических групп для нашего словаря:

Базовый словарь. Для каждого английского слова w были объединены синонимические группы $S^{l}_{C_w}$ для всех рассматриваемых языков $l \in L_{UWN}$. Синонимические группы которые относятся к разным частям речи слова w объединялись по отдельности друг от друга. В дальнейшем объединенные синонимические группы служили в качестве “межъязыковых” синонимических групп.

Модифицированный словарь (в дальнейшем: Top1). В рамках данного подхода модификации базового словаря, исходя из оценки их популярности в Princeton WordNet 3.1, некоторые смысловые концепции конкретных слов были отфильтрованы. Для каждого английского слова w, по отдельности для каждой из возможных частей речи данного слова, выбирались только самые частоиспользуемые смысловые концепты C^{Top1}_{w}. В итоговые “межъязыковые” синонимические группы объединялись только синонимические группы данного смыслового концепта $S^{l}_{C^{Top1}_{w}}$ для всех языков $l \in L_{UWN}$.

2.1.3.2.2. Дополнение словаря с использование машинного перевода

Для проверки качества словаря, полученного вышеописанным методом, была посчитана статистика его лексического покрытия текстов из 120,000 статей из Wikipedia. Данная статистика представлена на рисунок 2.4.

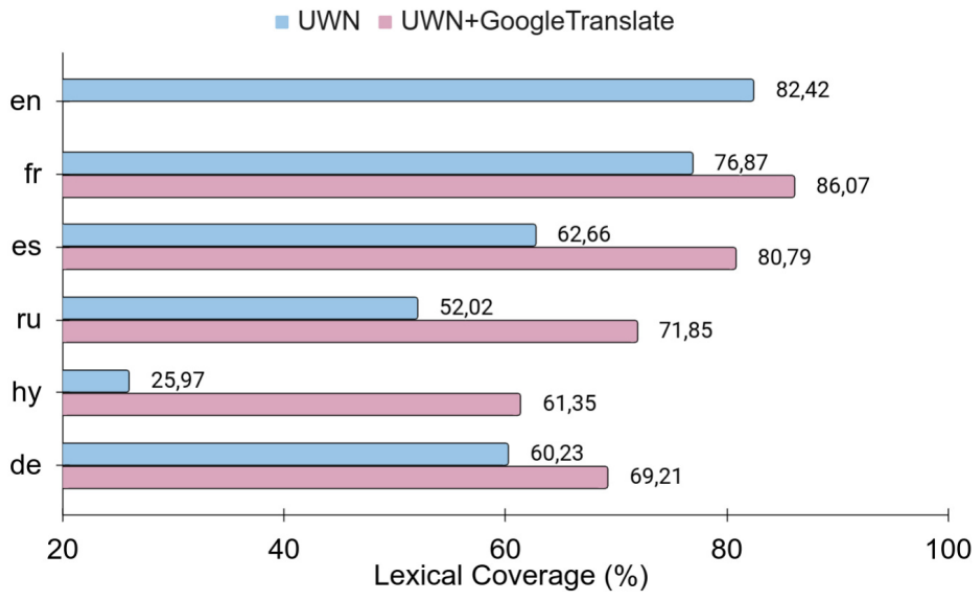

Рисунок 2.4 – Процент покрытия слов из 120 тысяч статей Wikipedia словарем UWN и его дополненной версией.

Исходя из представленной статистики, для некоторых смысловых концепций в словаре Universal WordNet не содержалось слов на некоторых из рассматриваемых языков. Во избежании данной проблемы, было решено искуственно дополнить полученный словарь.

Рисунок 2.5 – Схема дополнения кластеров UWN с использованием Google Translate.

Для тех “межъязыковых” синонимических групп, полученного ранее словаря, в которых не содержались слова на некотором языке, извлекались слова на английском языке и производился их перевод на недостающий язык. Для

дополнительного обогащения словаря перевод производился и на те языки которые присутствовали в “межъязыковых” синонимических группах.

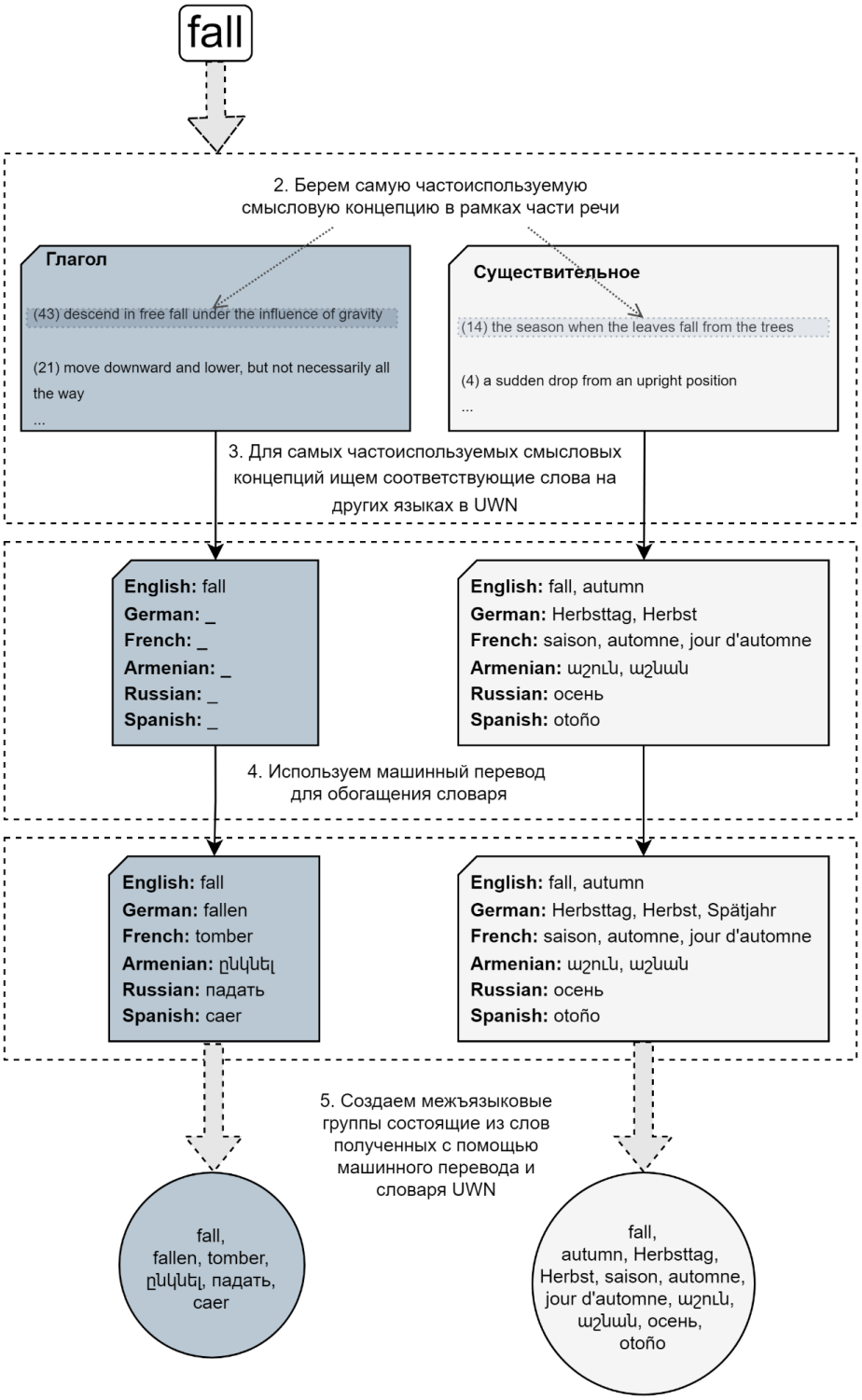

Рисунок 2.6 – Процесс создания первичного словаря синонимов с использованием тезауруса Universal WordNet, и процесс его дополнения с использованием машинного перевода.

Процесс перевода был осуществлен с использованием системы машинного перевода Google Translate[13]. Данный процесс представлен на рисунке 2.5**.** Перевод каждого слова производился с учетом части речи, которой принадлежит концепция рассматриваемой “межъязыковой” синонимической группы и соответственно все остальные слова данной группы.

Полученный таким образом словарь был назван “межъязыковым словарем синонимов” - “Cross-Language synonyms dictionary” (CL-SynDi).

Весь процесс получения словаря межъязыковых синонимов Top1 представлен на рисунке 2.6.

2.1.4. Эксперименты

2.1.4.1. Сравнение методов построения межъязыкового словаря синонимов

Для сравнения двух сгенерированных словарей межъязыковых синонимов (базовый словарь и Топ1) и оценки общей работы этапа извлечения фрагментов-кандидатов были произведены эксперименты на документах пары языков английский-русский.

В качестве документов-источников выступали 10,000 документов извлеченных из английской Wikipedia, используемые в рамках корпуса CrossLang [7]. В качестве анализируемых документов также выступали представленные в корпусе CrossLang 316 автоматически сгенерированных документов на русском языке. В каждом из 316 анализируемых документов содержались заимствованные фрагменты из 10,000 документов-источников.

Так как в процессе извлечения топ-К фрагментов-кандидатов достаточно что бы фрагмент из которого было произведено заимствование содержался в этих К

[13] https://translate.google.com/

фрагментах независимо от его позиции, то в качестве метрики для оценки работы этапа извлечения кандидатов была выбрана метрика Recall@K (2.9).

$$Recall@K = \frac{\text{Количество найденных релевантных фрагментов}}{\text{Количество всех релевантных фрагментов}} \quad (2.9)$$

Данная метрика показывает какой процент фрагментов из которых были произведены заимствования был найден после извлечения топ-К кандидатов для каждого из анализируемых фрагментов.

Для сравнения базового и модифицированного (Топ1) сгенерированных словарей межъязыковых синонимов, процесс индексации документов-источников, а также процесс поиска фрагментов-кандидатов производился по отдельности с учетом этих словарей. Результаты сравнения работы алгоритма поиска фрагментов-кандидатов с использованием различных словарей межъязыковых синонимов и при различных значениях K представлены в Таблице 2.

Таблица 2 – Результаты работы метода извлечения кандидатов с использованием разных словарей межъязыковых синонимов, при различных значениях K.

	Recall@K			
Словарь	K=1	K=5	K=10	K=50
Базовый	0,624	0,728	0,762	0,832
Топ1	**0,747**	**0,827**	**0,853**	**0,899**

Исходя из результатов показанных в таблице видно, что модифицированный словарь Топ1 основанный только на самых частоиспользуемых смысловых концептах показывает лучшие результаты. Это может быть связано с тем, что одинаковые слова встречаются в меньшем количестве синонимичных групп (т.е. используются в меньшем количестве смыслов) тем самым сужая возможный поиск, делая его более точным. Таким образом метод составления словаря

межъязыковых синонимов основанный на использовании только самых популярных смысловых концептах является лучшим.

Также, при использовании словаря Топ1 можно заметить, что даже при значении K равным единице, то есть при извлечении только одного самого близкого фрагмента-кандидата, находятся почти 75% от всех фрагментов из которых в действительности было произведено заимствование. При увеличении значения K до 50, количество найденных фрагментов достигает почти 90%.

После выявления лучшего метода составления словаря межъязыковых синонимов, было произведено сравнение двух словарей составленных на основе данного метода. Первый словарь был составлен только из слов полученных с помощью тезауруса Universal WordNet, второй словарь был дополнен поверх первого, как это было представлено в подразделе *2.1.3.2.2*. Сравнение было произведено с целью проверить эффективность дополнения словаря с помощью машинного перевода. Результаты сравнения двух словарей представлены в Таблице 3.

Таблица 3 – Результаты сравнения словаря основанного только на словах из UWN с словарем дополненным с помощью машинного перевода, на задаче извлечения фрагментов-кандидатов.

	Recall@K			
Словарь на основе	K=1	K=5	K=10	K=50
UWN	0,175	0,265	0,307	0,426
UWN + Google Translate	**0,747**	**0,827**	**0,853**	**0,899**

В таблице видно, что использование дополненного словаря приводит к результатам в разы превосходящим те, которые получаются при использовании словаря основанного только на UWN. При достижении в почти 75% значения Recall@K при использовании дополненного словаря с значением K=1, обычный словарь показывает плохие результаты в 17,5%. Таким образом, дополнение

словаря с использованием машинного перевода является эффективной операцией и приводит к значительным улучшениям результатов, что связано с недостатком слов в UWN для языков отличных от английского.

Дополнительно, для проверки устойчивости представляемого метода извлечения кандидатов к увеличению с течением времени количества документов проверочной базы, к уже использованным 10,000 документам итеративно добавлялись по 10,000 документов на каждом шагу из той же коллекции CrossLang. Документы добавлялись до момента, когда вся проверочная база коллекции CrossLang была бы проиндексирована. После каждой итерации добавления документов производилась оценка нахождения релевантных фрагментов для тех же 316 документов, что использовались ранее. Подробные результаты полученные при увеличении количества документов в проверочной коллекции представлены в Таблице 4.

Таблица 4 – Изменение значения оценки Recall@K при увеличении количества документов в проверочной коллекции, при разных значениях K.

		Recall@K							
Документов в базе	Фрагментов в базе	K=1	K=5	K=10	K=50	K=100	K=200	K=500	K=1000
10,000	499,190	0,75	0,83	0,85	0,90	0,91	0,93	0,94	0,95
20,000	987,865	0,72	0,81	0,84	0,89	0,91	0,92	0,94	0,95
30,000	1,485,945	0,70	0,79	0,82	0,87	0,89	0,91	0,93	0,94
40,000	1,977,502	0,68	0,78	0,81	0,87	0,89	0,90	0,92	0,93
60,000	2,963,917	0,66	0,77	0,80	0,85	0,88	0,89	0,91	0,93
80,000	3,940,512	0,64	0,76	0,79	0,85	0,87	0,89	0,91	0,92
103,000	5,112,228	0,63	0,75	0,78	0,84	0,86	0,88	0,90	0,92

Исходя из результатов показанных в таблице, можно заметить что, при увеличении значения K ухудшение оценки Recall@K происходит менее значимо,

то есть при увеличении базы в 10 раз, при значении, например, K=1000, оценка Recall@K опустилась на 3 процента.

2.1.4.2. Сравнение методов предобработки

Для выявления лучшего метода предобработки текста в задаче извлечения кандидатов были произведены эксперименты с 4 видами предобработок (во всех четырех методах слова были токенизированы, стоп-слова и пунктуационные символы были удалены, буквы были приведены к их строчной форме):

- Использование только лемматизации. (Lemma)
- Использование лемматизации и дополнение слов частями речи в которых они были использованы во фрагменте. (Lemma+POS)
- Использование лемматизации и обнаружение именованных сущностей в текстах с их последующей заменой. (Lemma+NER)
- Использование лемматизации, дополнение слов частями речи в которых они были использованы во фрагменте и обнаружение именованных сущностей в текстах с их последующей заменой. (Lemma+POS+NER)

Выполнение лемматизации слов, определение частей речи и именованных сущностей было произведено с использованием библиотеки Stanza[14] v1.2.1. [59, 60].

Stanza - это библиотека для обработки текста на разных языках. Она предоставляет инструменты для таких задач, как токенизация, лемматизация, синтаксический анализ, морфологический анализ текста и т.д..

Определение именованных сущностей в армянском языке. На момент проведения экспериментов в библиотеке stanza не существовало модели определяющей именованные сущности в текстах армянского языка. Исходя из этого, для армянского языка, с использованием архитектуры модели предоставляемой библиотекой stanza, была обучена модель обнаружения

[14] https://stanfordnlp.github.io/stanza/

именованных сущностей, которую можно было бы использовать через библиотеку stanza как в случае с другими языками.

Для создания модели был выбран набор данных ArmTDP-NER[15] . ArmTDP-NER это корпус обнаружения именованных сущностей на восточно-армянском языке содержащий в себе 1,949 текстовых фрагментов. Корпус ArmTDP-NER содержит 18 типов именованных сущностей: PERSON, NORP, FACILITY, ORGANIZATION, GPE, LOCATION, PRODUCT, EVENT, WORK OF ART, LAW, LANGUAGE, DATE, TIME, PERCENT, MONEY, QUANTITY, ORDINAL, CARDINAL.

Для решения поставленной задачи, были обучены разные модели для двух разных количеств используемых меток. В первом случае использовались все представленные 18 меток именованных сущностей, во втором случае были использованы только 4 метки именованных сущностей. Те слова, которые были размечены как “PERSON” и “ORGANIZATION” сохранили свои метки. Метки “GPE” и “LOCATION” были соединены в одну метку обозначающую местоположение. Оставшиеся 14 меток были приравнены к метке “MISC”, которая отвечала за разметку именованных сущностей не являющимися “PERSON”, “ORGANIZATION”, “GPE” или “LOCATION”.

Базовая архитектура модели предоставляемая библиотекой stanza основана на использовании предобученной модели CharLM [61], а также LSTM [62] слоев. Однако, в архитектуру базовой модели можно также добавить “слой” использующий для различных языков различные BERT-основанные модели. В рамках обучения модели, были опробованы обе архитектуры. Во втором случае, в качестве BERT-основанной модели использовалась XLM-RoBERTa [43], которая содержит в себе армянский язык.

Таким образом были обучены 4 модели, с использование двух разных количеств меток именованных сущностей и двух разных архитектур моделей. Результаты полученные для 4 моделей на тестовой выборке ArmTDP-NER, представлены в Таблице 5**.**

[15] https://github.com/myavrum/ArmTDP-NER/

Таблица 5 – Результаты полученные моделями определения именованных сущностей в текстах армянского языка (в таблице представлено значение оценки F1).

Модель	**4 меток**	**18 меток**
CharLM	86,28	86,68
CharLM + XLM-R	89,85	89,31

Исходя из результатов показанных в Таблице 5 лучшие результаты были получены при использовании 4 меток именованных сущностей и с использованием модели XLM-RoBERTa.

В дальнейшем в экспериментах использовались модели обученные для предсказания 4 меток.

Определение лучшего метода предобработки фрагментов. Эксперименты для сравнения представляемых четырех методов предобработки фрагментов и выбора лучшего из них, были также проведены на документах пары английский-русский с использованием такой же части в 10,000 документов корпуса CrossLang как и в *2.1.4.1*.

Таблица 6 – Сравнение различных методов предобработки фрагментов в задаче извлечения фрагментов-кандидатов.

		Recall@K							
Словарь	Предобработка	K=1	K=5	K=10	K=50	K=100	K=200	K=500	K=1000
Базовый	Lemma	0,62	0,73	0,76	0,83	0,86	0,88	0,90	0,92
Базовый	Lemma+POS	0,39	0,50	0,54	0,63	0,68	0,71	0,76	0,80
Базовый	Lemma+NER	0,47	0,59	0,63	0,72	0,76	0,79	0,83	0,86
Базовый	Lemma+POS+NER	0,32	0,43	0,47	0,58	0,62	0,67	0,72	0,77
Топ1	**Lemma**	**0,75**	**0,83**	**0,85**	**0,90**	**0,91**	**0,93**	**0,94**	**0,95**
Топ1	Lemma+POS	0,52	0,62	0,66	0,75	0,78	0,81	0,84	0,87
Топ1	Lemma+NER	0,64	0,73	0,76	0,83	0,86	0,88	0,90	0,92
Топ1	Lemma+POS+NER	0,46	0,58	0,62	0,71	0,75	0,78	0,82	0,85

Каждый из четырех методов дополнительно был проверен с каждым из типов межъязыковых словарей (базовый и Топ1). Результаты полученные для этапа извлечения кандидатов представлены в Таблице 6.

Исходя из полученных результатов, можно сказать, что модификации классического использования лемматизации, в виде добавления к словам их меток частей речи или замены слов на метки именованных сущностей, не дали положительных результатов. Соответственно в качестве метода предобработки фрагментов был выбран метод с использованием только лемматизации, что в свою очередь является также менее ресурсозатратным процессом.

2.1.4.3. Детали реализации

Описанный в данной главе алгоритм извлечения фрагментов-кандидатов, был реализован с использованием инструмента Apache Solr[16].

Apache Solr - это платформа полнотекстового поиска. Поиск в Solr работает на основе обратного индекса, и все документы хранящиеся в его базе заранее индексируются. После обнаружения релевантных документов или фрагментов текста, Solr производит их сортировку используя заранее обозначенную функцию подсчета близости между двумя текстами, также называемая функцией ранжирования. В данной работе в качестве такой функции была выбрана функция Okapi BM25. В конце Solr возвращает отсортированный, исходя из значений полученных с использованием функции Okapi BM25, список фрагментов-кандидатов.

Solr используется для поиска релевантных документов в текстах одного языка, и для его применения в качестве межъязыкового поисковика, была использована встроенная в Solr функция улучшения поиска при помощи, заранее определенного, словаря синонимов, где индексация производится с учетом кластеров синонимов данного словаря. Для использовании данной функции в

[16] https://solr.apache.org/

нуждах межъязыкового поиска, словарь синонимов был интерпретирован как словарь межъязыковых синонимов, и в качестве такого словаря использовался словарь Топ1, описанный в *2.1.3.2*.

В качестве гиперпараметров используемых в функцие Okapi BM25 использовались *k1 = 1.2* и *b = 0.75*.

В итоге значение K было выбрано равным 50, для обеспечения высокой полноты, при этом не оставляя для дорогостоящего этапа детального анализа очень большое количество фрагментов для обработки. В работе [63], которая описывает процесс нахождения межъязыковых заимствований между английским и венгерским языками, также на первом этапе извлечения кандидатов брались от 10 до 50 кандидатов.

2.1.5. Методы обработки текстов

Данный подраздел посвящен различным методам обработки текстов нацеленным на улучшение процесса извлечения кандидатов, с помощью уменьшения количества ненужной для поиска информации, с добавлением дополнительной информации, как части речи для улучшения поиска, а также предварительного сокращения количества документов в проверочной коллекции документов.

2.1.5.1. Токенизация

Токенизация - это процесс разбиения текста на меньшие единицы, такие единицы называются токенами. Токенизация может включать в себя такие процессы, как разбиения текста по предложениям, по словам, по лексемам, по буквам. Токенизация является одним из первых этапов в процессе обработки

текста. Этот процесс позволяет обрабатывать и анализировать текст на уровне слов и грамматических конструкций.

2.1.5.2. Удаление стоп-слова

Стоп-слова - это часто употребляемые слова, которые обычно не несут никакой семантической нагрузки и не добавляют новой информации в текст. Процесс удаление стоп-слов из текста один из базовых процессов предобработки текстов и может быть полезен для улучшения эффективности некоторых алгоритмов машинного обучения, за счет уменьшения количества слов в рассматриваемых текстах и их упрощения.

В процессе нахождения кандидатов, при оценке схожести различных фрагментов текстов и их ранжировании используется функция Okapi BM25, в которой присутствует зависимость от общего количества токенов и от количества одинаковых токенов в двух сверяемых текстовых фрагментах. Данная зависимость приводит к тому, что при наличии стоп-слов в рассматриваемых фрагментах значение выдаваемое функцией будет учитывать совпадение слов которые не содержат никакого смысла, а также влияют на длину предложения.

2.1.5.3. Лемматизация

Лемматизация - это процесс приведения слова к его словарной форме (лемме), то есть к начальной форме слова, которая не изменяется при согласовании с другими словами или при образовании форм слова. Лемматизация необходима для улучшения качества анализа текстов и для более точной идентификации слов в тексте. Она позволяет убрать отличия между разными формами слова, такими как единственное и множественное число, прошедшее и настоящее время, и т.д., тем самым уменьшая разреженность данных.

Процесс нахождения кандидатов основан на использовании межъязыкового

словаря межъязыковых “синонимов”, который в свою очередь содержит в себе синонимичные группы в которых содержаться слова в их словарной форме, что позволяет значительно сократить количество слов в одной группе. Таким образом, для увеличения эффективности словаря межъязыковых “синонимов” и анализируемые документы, и документы-источники проходят через процесс лемматизации.

2.1.5.4. Определение частей речи

Определение части речи - это процесс определения того, к какой категории (части речи) относится слово в тексте. Данный процесс играет существенную роль в определении смысла слова для омонимичных слов.

В рамках этапа нахождения первичных документов-кандидатов, основанного на словаре межъязыковых “синонимов” в котором не содержится информации о значении слова, определение частей речи может помочь в дополнительном разграничении слов.

Таким образом, к каждому слову каждого кластера словаря межъязыковых “синонимов” добавлялась метка части речи к которой относятся слова конкретного кластера. Каждый из кластеров в словаре является каким-то смысловым концептом в рамках которого и объединены слова на разных языках. Для каждого смыслового концепта в Universal WordNet содержалась часть речи данного концепта. Таким образом, к каждому слову каждого кластера прикреплялась метка части речи смыслового концепта, которая представлена в UWN.

Также, при добавлении документов-источников в базу, тексты документов проходили этап определения частей речи и к каждому из слов данного текста добавлялась метка соответствующей части речи. В процессе заполнения базы документов-источников, слова подаваемых документов индексировались с учетом

частей речи и соответственно заменялись на индексы кластеров в которых они содержались с соответствующей меткой части речи.

Аналогично, все слова анализируемых документ перед прохождением этапа поиска документов-кандидатов получали собственные метки частей речи. Процесс поиска производился с использованием размеченного по частям речи словаря межъязыковых “синонимов”.

Для определения частей речи слов анализируемых документов и документов-источников также использовалась библиотека stanza.

Таким образом, используя в процессе первичного поиска метки частей речи, была произведена попытка решить проблему неоднозначности слов.

2.1.5.5. Определение именованных сущностей

Именованная сущность - это слово или словосочетание принадлежащее к группе определенных объектов или к группе имеющей некие общие черты. К именованным сущностям относятся, например, географические локации, имена людей, наименования организаций, и т. д.. Задача определения именованных сущностей ставится, как извлечение подобных слов или словосочетаний из текстов.

Идея использования модели определения именованных сущностей в рамках задачи обнаружения межъязыковых заимствований пришла от понимания того, что для именованных сущностей написанных на двух разных языках может возникнуть проблема с приведением их к одному виду, что приведет к ухудшению результатов поиска кандидатов. Для разрешения данной проблемы и была произведена попытка использования подобных моделей.

Фрагменты-источники и анализируемые фрагменты проходили через модель определения именованных сущностей, и каждое слово фрагмента получало свою метку. В качестве меток для именованных сущностей использовались следующие 4 метки:

- “PER” - метка отвечающая за личностей, персонажей и т.д.
- “LOC” - метка отвечающая за местоположения
- “ORG” - метка отвечающая за организации
- “O” - метка отвечающая за все остальные слова не являющимися именованными сущностями

Те слова или словосочетания которые были размечены моделью, как “PER”, “LOC” или “ORG” заменялись соответствующей меткой данного типа. В дополнение проводилась нумерация каждой из меток одного типа: рядом с каждой меткой добавлялся индекс указывающий на то, какая по счету во фрагменте метка конкретного типа. Такая нумерация производилась во избежании искусственного увеличения оценки близости текстов в ситуации, где все метки одного типа воспринимались бы, как одно и то же слово.

2.1.5.6. Фильтрация диалектов при обработке документов

Многие языки имеют различные диалекты и формы, которые отличаются между собой синтаксически, морфологически и фонетически. Тексты различных диалектов могут храниться наравне с другими в проверочных наборах данных тем самым увеличивая их размер и влияя на качество обнаружения заимствований. Для решения данной проблемы, в рамках данного подраздела на примере диалектов армянского языка решается задача классификации и последующей фильтрации текстов написанных на диалектах представленная в [6].

Рассматриваются 3 варианта армянского языка: Восточный, Западный, и Классический. Задача классификации диалектов схожа с задачей классификации языка, с некоторыми отличиями усложняющими поставленную задачу. Так, например, диалекты одного языка используют ту же символику (буквы) и имеют пересечения в словаре используемых слов.

Существуют лексико-основанные подходы решающие задачу классификации языков на основе стоп-слов и диактрических знаков [64]. В

Армянском языке нет диактрических символов, а в случае с диалектами стоп-слова могут быть одинаковыми для разных диалектов, таким образом наравне со способом использующим только стоп-слова был опробован его модифицированный вариант с дополнением стоп-слов самыми частовстречаемыми словами. Также, есть методы решающие задачу классификации языков с использованием нейронных сетей. Самой популярной имплементацией подобного метода является библиотека fastText[17].

Для решения задачи классификации диалектов, были опробованы 3 метода:

Метод основанный на стоп-словах. Пусть W_d словарь стоп-слов диалекта d ($d \in$ {Восточный, Западный, Классический}). W_{e} - множество слов содержащихся в тексте E. Для каждого текста E, предсказание диалекта производилось по следующей формуле:

$$label(E) = argmax_d(|W_d \cap W_{\text{e}}|) \quad (2.10)$$

Стоп-слова для западного и классического армянского языка были собраны вручную, а для восточного армянского были взяты из онлайн ресурса[18].

Лексико-основанный метод. В данном случае для каждого из диалектов был составлен собственный словарь. Составление данных словарей производилось с использование восточно-армянской и западно-армянской Wikipedia[19] [20], а для классического армянского была использована библиотека Digilib[21]. На каждом из наборов данных была подсчитана частота слов. Пусть $V_{A,k}$ множество топ-k самых частовстречаемых слов в диалекте A, финальный словарь для диалекта A составлялся следующим образом:

[17] https://fasttext.cc/blog/2017/10/02/blog-post.html
[18] https://github.com/stopwords-iso/stopwords-hy
[19] https://hy.wikipedia.org/wiki/%D4%B3%D5%AC%D5%AD%D5%A1%D5%BE%D5%B8%D6%80_%D5%A7%D5%BB
[20] https://hyw.wikipedia.org/wiki/%D4%B3%D5%AC%D5%AD%D5%A1%D6%82%D5%B8%D6%80_%D4%B7%D5%BB
[21] https://digilib.aua.am/am/about/about

$$D_A = V_{A,k} \setminus (V_{B,k} \cup V_{C,k}), \quad (2.11)$$

где $V_{B,k}$, $V_{C,k}$ множества топ-k самых частовстречаемых слов в диалектах B и C.

Метод основанный на нейронных сетях. В рамках данного метода использовалась модель классификации языков fastText представленная компанией Facebook. Данная модель основана на использовании обучаемых векторов n-грамм слов их усреднении и использовании линейного классификатора. Те же данные что и для лексико-основанного метода были использованы для обучения данной модели классификации диалектов.

Для оценки представляемых методов были созданы два тестовых набора на основе восточно-армянской и западно-армянской Wikipedia и библиотеки Digilib. Первый набор состоял из 1500 предложений, по 500 на каждый из диалектов, где средняя длина предложения составляла 18 слов или ≈130 символов. Второй набор состоял из полных текстов по 100 текстов на каждый из диалектов, где средняя длина текста составляла ≈600 слов или ≈4150 символов.

Для лексико-основанного метода были опробованы несколько значений k самых частовстречаемых слов, в итоге было выбрано значение $k = 10000$.

Для метода основанного на нейронной сети, было обучено несколько моделей fastText на разном количестве обучающих примеров. Также, для каждого количества обучающих примеров, в качестве начальных векторов n-грамм слов использовались два варианта векторов: предобученные вектора предоставляемые fastText[22] размерностью 300, и вектора с размерностью 16. Результаты показанные разными моделями, с разными векторами и количеством обучающих примеров представлены на рисунке 2.7.

[22] https://fasttext.cc/docs/en/crawl-vectors.html

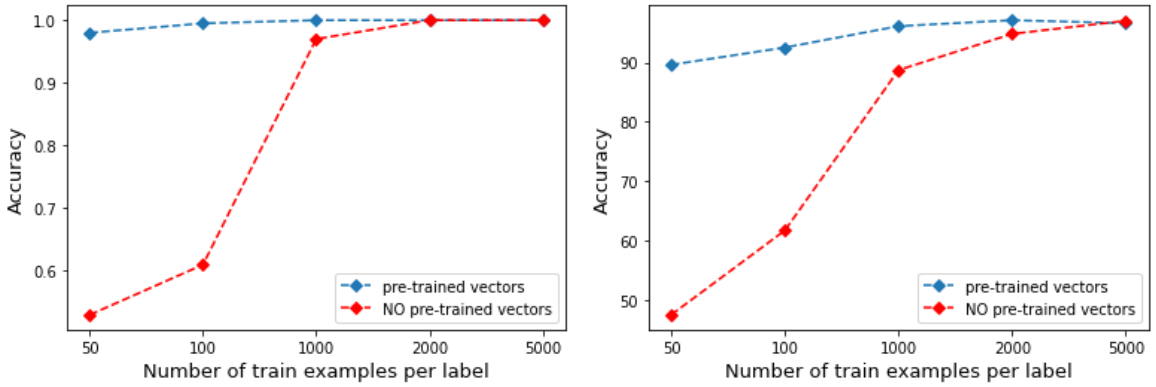

Рисунок 2.7 – Результаты различных моделей на тестовой выборке с текстами (слева) и на тестовой выборке с предложениями (справа).

Дополнительно, для минимизации затраты времени на определение диалекта на уровне полных текстов, была опробована подача фиксированного количества первых n символов на вход модели. Значение n менялось от 10 до 200 символов. Результаты подсчитаны для лучшей модели полученной на предыдущем шагу и показаны на рисунке 2.8.

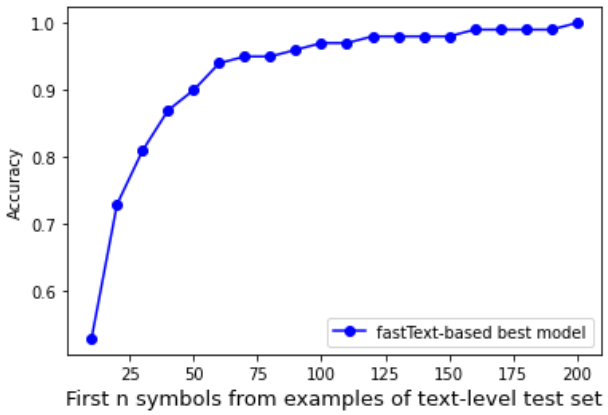

Рисунок 2.8 – Зависимость точности классификации диалектов от количества подаваемых на вход модели символов.

В результате при подачи 200 символов модель достигает такой же точности, что и при подаче всего документа, но в то же время работает примерно в 20 раз быстрее.

Сравнение трех методов представлено в Таблице 7.

Таблица 7 – Сравнение трех представляемых методов классификации диалектов армянского языка.

	Тестовый набор с предложениями	**Тестовый набор с текстами**
	Доля правильных ответов	Доля правильных ответов
Стоп-слова	0.51	0.55
Лексико-основанный	0.63	0.67
Нейронная Сеть	0.98	1.00

Для понимания важности фильтрации текстов написанных на диалектах используя модель fastText была произведена классификация текстов существующих популярных наборов данных Wikisource[23] и mC4 [65]. Количество текстов разбитых по диалектам в результате классификации представлены в Таблице 8.

Таблица 8 – Количество текстов полученных после процесса классификации для каждого из рассматриваемых диалектов армянского языка.

	Восточный	**Западный**	**Классический**
mC4	2,273,405	99,844	13,796
Wikisource	84,398	9,440	8,044

Таким образом, при фильтрации текстов западного и классического диалектов проверочные наборы данных уменьшаются на 5% и 17% для mC4 и Wikisource соответственно.

2.1.6. Выводы

В рамках данного раздела был представлен алгоритм извлечения текстовых фрагментов-кандидатов реализованный с помощью инструмента Apache Solr и с использованием представления слов в независимой от языка форме.

[23] https://wikisource.org/wiki/Main_Page

Были протестированы два метода представления слов в независимой от языка форме с использованием векторных представлений слов, а также мультиязычных тезаурусов. Исходя из результатов полученных в рамках тестирования методов представления слов в независимой от языка форме, в качестве метода для дальнейшего использования был выбран метод основанный на использовании мультиязычного тезауруса Universal WordNet.

Были проведены экспериментальные сравнения различных методов получения итогового словаря межъязыковых синонимов с использованием UWN. Лучше всего себя показал словарь основанный на использовании только самых популярных смысловых концептов и дополненный с помощью машинного перевода.

Были проведены исследования различных функций оценки близости текстов. Наиболее подходящей функцией для применения в поиске фрагментов-кандидатов была выбрана функция Okapi BM25, которая является гибкой относительно смены количества документов в проверочной коллекции.

Была создана модель классификации диалектов армянского языка, позволяющая отфильтровывать ненужные диалекты тем самым уменьшая проверочную базу данных.

2.2. Детальный анализ

В данном разделе рассматривается второй этап обнаружения межъязыковых заимствований - детальный анализ. Данный этап предназначен для фильтрации кандидатов полученных после окончания первого этапа и сопоставления конкретных фрагментов текстов анализируемого документа с конкретными фрагментами документов-кандидатов.

В первом подразделе описывается алгоритм детального анализа используемый в рамках представляемого метода. Во втором подразделе описывается процесс выбора языковой модели и ее обучения, а также описывается

метод автоматической генерации сложной тестовой выборки для задачи определения является ли одно предложение переводом другого. В третьем подразделе представлен способ применения алгоритма детального анализа. Четвертый подраздел посвящен искусственным атакам на языковые модели, в том числе и модель детального анализа. В рамках четвертого подраздела также представляется новый метод генерации искусственных атак на языковые модели обходящий по доле успешности атак все существующие методы. В пятом подразделе представляется методика выбора модели для этапа детального анализа с учетом угрозы быть подвергнутым искусственным атакам.

2.2.1. Описание алгоритма

В качестве метода для произведения детального анализа был выбран метод попарного сравнения предложений анализируемого фрагмента со всеми предложениями топ-К фрагментов-кандидатов, который был представлен в работе [42]. В рамках процесса попарного сравнения этих предложений для каждой пары производилась бинарная классификация – являются ли предложения данной пары переводом друг друга. Процесс классификации производился с использованием многоязычной языковой модели нейронной сети. Работа модели представлена на рисунке 2.8.

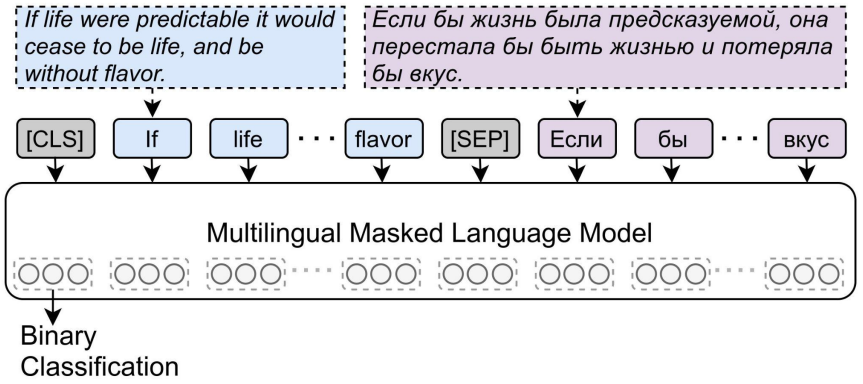

Рисунок 2.9 – Работа модели бинарной классификации является ли одно предложение переводом другого, которая используется на этапе детального анализа.

Если для определенного предложения из анализируемого документа процесс классификации выдает несколько возможных предложений-переводов, в таком случае переводом считается только то предложение-кандидат в паре с которым модель выдала наибольшее значение.

2.2.2. Эксперименты

2.2.2.1. Выбор языковой модели

Для реализации процесса бинарной классификации было решено использовать многоязычные языковые модели основанные на архитектуре “трансформер”. “Трансформер”-основанные модели достигают лучших результатов в различных задачах межъязыковой обработки текстов, и в частности в задаче обнаружения перефразирований между двумя предложениями [66-68], которая является смежной с задачей определения перевода. В рамках данного подраздела описывается процесс сравнения различных языковых моделей на задаче классификации перевода между двумя предложениями представленный в [5].

При выборе моделей для их последующего сравнения, были поставлены два ограничения:

- Модель должна была состоять из меньше чем 300 миллионов параметров, что дает некое ограничение сверху на скорость работы модели.
- Модель должна была содержать в себе Армянский язык, так как алгоритм поиска межъязыковых заимствований должен будет быть использован для нахождения заимствований в текстах армянского языка.

С учетом данных ограничений для сравнения были отобраны 6 моделей, которые представлены вместе с количеством их параметров в Таблице 9.

Таблица 9 – Отобранные для последующего сравнения модели и количество их параметров.

Модели	**Количество параметров**
mBERT[24][69]	172M
mDistilBERT[25][70]	134M
XLM-RoBERTa[26][43]	270M
SBERT Multilingual MiniLM-L12[27][71, 72]	117M
SBERT Multilingual MPNet[28] [71, 72]	278M
SBERT Distiluse Multilingual[29] [71, 72]	135M

В рамках сравнения моделей, должны были быть достигнуты следующие цели:

- Оценка быстродействености каждой из моделей
- Определение лучшей модели с точки зрения точности классификации
- Определение межъязыковой переносимости знаний моделей, т.е. определение того, будет ли более эффективно использование одной модели для различных языков или использование одной модели для языков в рамках одной языковой подгруппы, или все же использование по одной модели для каждого языка.

2.2.2.1.1. Данные

Для достижения поставленных целей, в качестве языков для произведения экспериментов, были выбраны 10 языков индо-европейской языковой семьи:

[24] https://huggingface.co/bert-base-multilingual-cased
[25] https://huggingface.co/distilbert-base-multilingual-cased
[26] https://huggingface.co/jplu/tf-xlm-roberta-base
[27] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
[28] https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2
[29] https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2

армянский, русский, английский, французский, немецкий, сербский, шведский, голландский, испанский, чешский. Данные языки в свою очередь также разбиты по 4 языковым группам: германская, романская, славянская и армянская языковые группы. В дополнение к разделению языков по 4 группам, также внутри каждой такой группы языки были выбраны таким образом, что два языка внутри одной группы были ближе к друг другу чем к третьему языку той же группы (данное разделение не было учтено для армянской языковой группы, так как в нее входит только один язык). Таким образом языки были выбраны со следующим разбиением:

- Романская группа
    - Иберо-Романская подгруппа
        - Испанский язык
        - Португальский язык
    - Галло-Романская подгруппа
        - Французский язык
- Славянская группа
    - Кириллица
        - Русский язык
        - Сербский язык
    - Латиница
        - Чешский язык
- Германская группа
    - Западногерманская подгруппа
        - Немецкий язык
        - Голландский язык
    - Северо-Германская подгруппа
        - Шведский язык
- Армянская группа
    - Армянский язык

В случае с романской и германской группами, языки были выбраны исходя из подгрупп к которым они принадлежат. В случае со славянской группой, языки были выбраны исходя из их алфавитной близости, т.е. в русском и сербском используется кириллица, а в чешском латиница.

Обучающие данные. В качестве данных для обучения моделей определения перевода, и последующего выбора лучшей из них, была попытка использования существующего набора данных Wikimatrix, а также был сгенерирован собственный набор.

Wikimatrix. В качестве данных для обучения, изначально было решено извлечь и использовать данные из корпуса WikiMatrix [73]. Корпус WikiMatrix - это корпус параллельных текстов, содержащий в себе 135 миллионов пар параллельных предложений из Wikipedia для 85 языков. Предложения распределены между 1620 пар языков.

Из полученного множества пар, учитывая 6 рассматриваемых языков, были отобраны 30 всевозможных. Файлы каждой из пар языков содержат в себе некоторое количество пар предложений.

Корпус Wikimatrix является автоматически сгенерированным и содержит в себе некорректные пары предложений. Фильтрация подобных пар, для каждого из 30 файлов пар языков, была воспроизведена в следующие 3 этапа:

- Удаление стоп слов и последующая фильтрация предложений длиной меньше 5 и больше 30.
- Распознавание языка, для того чтобы отсеять те пары предложений, в которых используется язык отличный от языков рассматриваемого файла. Распознавание производилось с использованием двух моделей определения языка: fastText[30] и polyglot[31]. Были оставлены только те пары предложений, для которых обе модели предсказали одинаковый, совпадающий с языками файла, язык.

[30] https://fasttext.cc/blog/2017/10/02/blog-post.html
[31] https://polyglot.readthedocs.io/en/latest/index.html

- С использование модели определения степени корректности перевода ModelFront[32], все те пары предложений, для которых модель показывала высокий риск неправильного перевода (>50%) были также удалены (Таблица 10).

Таблица 10 – Примеры предложений которые были отброшены (сверху) и подтверждены (снизу) с использованием инструмента ModelFront.

Предложение 1	**Предложение 2**	**Процент Риска**
By the time they were discovered they had already cracked 47642 passwords.	Ժամանակի հետ դրանք հայտնաբերվեցին բայց կոտրվել էին 47642 գաղտնաբառ:	55,98%
List of Olympic medalists in men's figure skating photos and autographs.	Գեղասահքի օլիմպիական խաղերում մեդալակիրների ցուցակը:	45,89%

После вышеупомянутых трех этапов фильтрации, для некоторых пар языков (в частности пары в которых содержался армянский язык) качество пар предложений по прежнему оставалось низким, таким образом от такого рода данных было решено отказаться.

Сгенерированная обучающая выборка. После неудачной попытки использования готовых параллельных данных WikiMatrix, было принято решение генерировать собственную обучающую выборку. Для получения положительных примеров из английской Wikipedia случайным образом были извлечены 5,000 предложений, которые содержали в себе более 5 слов. Данные предложения затем были переведены на все 10 рассматриваемых языков с использованием инструмента машинного перевода Google Translate. После чего переведенные предложения были соединены в пару со своими английскими аналогами.

Для получения отрицательных примеров, тем же образом, что и для положительных примеров из английской Wikipedia были извлечены 400,000 предложений. После извлечения, данные предложения прошли процесс предобработки, где из них были удалены стоп-слова, пунктуационные символы, и

[32] https://www.modelfront.com/

слова которые содержали в себе цифры. Предобработанные 400,000 предложений были затем разбиты на две группы по 200,000. Предложения из одной группы были затем поочередно сравнены с предложениями из второй группы. В качестве функции оценки близости выступала функция MinHash. После чего, случайным образом были отобраны 5,000 пар близость которых была на уровне от 40% до 80% по функции MinHash. Дополнительно, была произведена проверка что бы внутри отобранных случайным образом 5,000 пар не содержалось одинаковых предложений. Таким образом, после процесса сравнения и извлечения были получены 5,000 пар предложений, где оба предложения были написаны на английском языке. Одно из предложений каждой из пар было переведено на все рассматриваемые 10 языков, с использованием инструмента Google Translate.

В итоге, исходя из описанных языковых групп, а также для достижения поставленных целей, были сгенерированы 14 различных обучающих набора. По одному набору для каждого из рассматриваемых языков по отдельности в паре с английским, по одному набору для каждой из рассматриваемых языковых групп (кроме армянской группы) и набор в котором содержались пары предложений на всех 10 языках. Эти наборы содержали в себе по 5,000 положительных и отрицательных примеров, причем количество примеров содержащих тот или иной язык было равным. Таким образом 5,000 положительных и 5,000 отрицательных примеров разбивались исходя из того сколько языков используется в конкретном наборе, т. е., например, в наборе в котором содержались примеры для всех 10-ти языков содержалось по 500 положительных и 500 отрицательных примеров для каждого из языков, таким же образом были разбиты и наборы для языковых групп, где используются по 3 языка.

Тестовые данные. В качестве данных на которых производилось бы сравнение отобранных моделей, в контексте детального анализа, использовались два существующих набора данных Negative-1 и Negative-4[33] (в дальнейшем Neg-1 и Neg-4) для пары языков русско-английский. Так как, для многих языков не существовало тестовых наборов для подобной задачи, проверка работы модели

[33] http://nlp.isa.ru/ru-en-text-align-corp/

для других языков, в паре с английским, была произведена с использованием автоматически сгенерированных тестовых корпусов, основанных на корпусе перефразирований Microsoft Research Paraphrase Corpus[34] [74] (в дальнейшем MRPC). Которые в свою очередь в отличии от Neg-1 и Neg-4 содержат в себе не только негативные примеры пары предложений которых близки друг к другу с лексической точки зрения, но и примеры в которых пары предложений далеки с лексической точки зрения, но близки с семантической.

Negative-1, Negative-4. Данные два корпуса содержат в себе примеры только англйско-русских пар предложений. В обоих корпусах в качестве примеров параллельных предложений содержатся пары предложений из параллельного корпуса Яндекс[35] [75] в количестве 16,000, а также переведенные вручную 4,000 пары. Данные пары были использованы в качестве позитивных примеров. Для получения же негативных примеров каждое предложение на русском языке из 20,000 позитивных сравнивалось со всеми предложениями на английском языке (исключая то, которое составляло с ним пару). Сравнение производилось с использованием различных метрик подсчета межъязыковой близости между двумя предложениями. Далее для каждого предложения на русском языке к нему в пару добавлялось предложение которое, исходя из различных метрик, являлось самым близким на английском языке. Однако, в таком случае может возникнуть ситуация, где для нескольких предложений на русском языке самое близкое к ним предложение одно и то же. В таких случаях самое близкое предложение присваивалось только одному предложению на русском, остальным присваивались их последующие ближайшие предложения на английском. Логика такого подхода в получении негативных примеров заключается в том, что использование случайных пар предложений (в которых не будет пересечений) в качестве негативных примеров не смоделировали бы реальную ситуацию возникающую на этапе детального анализа, на котором приходится фильтровать переведенные предложения от тех которые имеют некую схожесть между собой.

[34] https://www.microsoft.com/en-us/download/details.aspx?id=52398
[35] https://translate.yandex.ru/corpus?lang=en

Для корпуса Neg-1 брали только один негативный пример с ближайшим предложением на английском. Для Neg-4 же брали топ-4 ближайших предложения на английском, соответственно получая по 4 негативных примера для каждого предложения на русском языке, что исходило из логики того, что в реальности примеров которые не являются плагиатом на много больше тех, которые им являются. Итоговое количество примеров данных двух наборов представлены в Таблице 11.

Таблица 11 – Количество позитивных и негативных примеров для каждой из выборок Negative-1 и Negative-4

	Обучающаю			**Валидационная**			**Тестовая**		
	Все	Поз.	Нег.	Все	Поз.	Нег.	Все	Поз.	Нег.
Neg-1	28,319	14,167	14,152	3,999	2,000	1,999	7,998	4,000	3,998
Neg-4	65,961	14,167	51,794	9,265	2,000	7,265	18,613	4,000	14,613

Генерация собственной сложной тестовой выборки. В основном, существующие наборы данных для задачи детального анализа при генерации негативных примеров основываются на оценке лексической близости. Однако, при использовании подобного подхода, не ставятся в учет те примеры которые могут быть семантически близкими друг другу, но при этом лексически не иметь никакой близости. Исходя из того, что задача классификации, является ли одно предложение переводом другого, близка к задаче определения перефразирований между парой предложений (перефразирование - это фактически перевод в тот же язык), то возникла идея использования корпуса Microsoft Research Paraphrase Corpus (MRPC) для генерации тестовой выборки для нашей задачи (Таблица 12).

Таблица 12 – Количество примеров в классическом тестовом корпусе MRPC.

	Общее число примеров	**Перефразирования**	**Не Перефразирования**
MRPC	1725	1147	578

В рамках данного корпуса содержатся примеры отмеченные меткой “*не перефразирование*”, которые семантически близки друг другу и при этом лексически далеки. Тестовые данные были сгенерированы для 10 языков в паре с английским с использованием машинного перевода поверх корпуса MRPC.

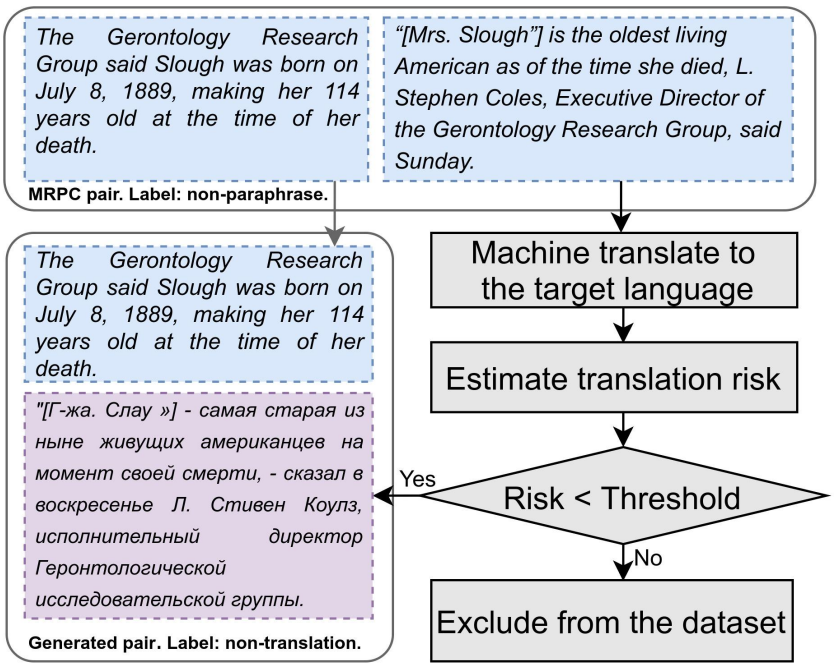

Рисунок 2.10 – Процесс генерации сложной тестовой выборки на основе перевода предложений из набора перефразирований MRPC.

Процесс генерации тестовых данных на основе MRPC описан на рисунке 2.10. Для каждой пары предложений из корпуса, одно из этих предложений (всегда первое) переводится, с помощью Google Translate, на один из 10 рассматриваемых языков помимо английского. Далее каждое переведенное предложение вступает в пару с предложением которое являлось парой для его оригинала. Однако, прежде чем вступать в пару с соответствующими предложениями на английском, во избежании получения примеров на которые влияла точность машинного перевода, переведенное предложение вместе с его оригиналом подавалось в систему оценки рисков перевода ModelFront. Данная система принимает на вход два параллельных текста, на выходе выдает процент риска того, что перевод был произведен плохо. Те пары, для которых система

ModelFront выдавала процент риска выше 50%, были отфильтрованы и не входили в конечные тестовые выборки.

В итоге, для тех пар предложений который прошли фильтрацию средний процент риска варьировался от 7% до 35% в зависимости от языка (рисунок 2.11).

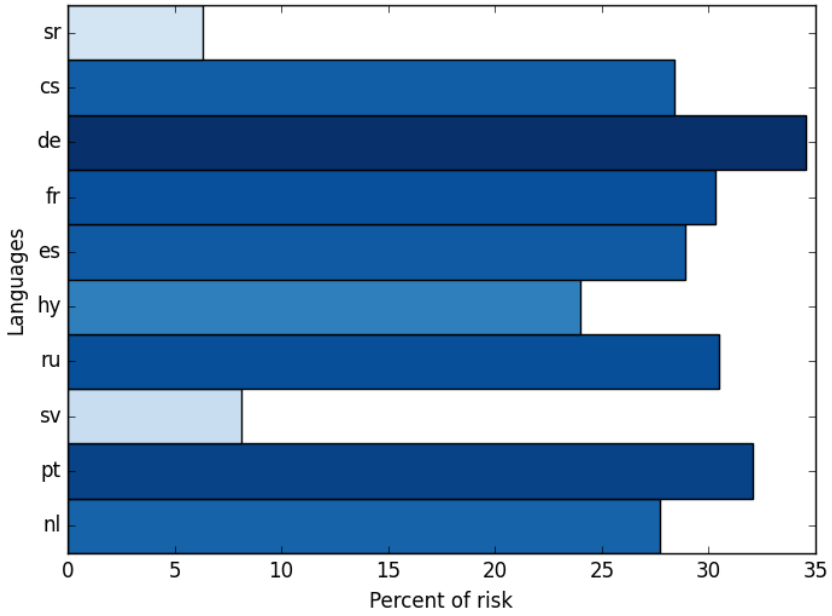

Рисунок 2.11 – Средний процент риска правильного перевода для включенных в итоговую выборку примеров.

Данный подход позволил нам без использования человеческих ресурсов, как это было сделано, например, для генерации корпуса SemEval-17 Task 1 [76], получить тестовые примеры для множества языков, в том числе и для такого малоресурсного языка как армянский. Итоговое количество примеров для каждой из используемых тестовых выборок представлено в Таблице 13.

Таблица 13 – Количество примеров для каждой из используемых тестовых выборок

Набор Данных	Языки	Число примеров	Положительные	Отрицательные
Сгенерированные на основе MRPC	EN - ES	1457	977	480
	EN - PT	1526	1009	517
	EN - FR	1397	916	481
	EN - RU	1250	843	407
	EN - SR	1644	1089	555
	EN - CS	629	424	205
	EN - DE	1154	768	386
	EN - NL	1268	848	420
	EN - SV	1665	1105	560
	EN - HY	901	622	279
Negative-1	EN - RU	7998	4000	3998
Negative-4	EN - RU	18613	4000	14613

С учетом того, что при создании данных корпусов использовался корпус MRPC в котором содержались семантически близкие негативные примеры *“не перефразирование*”, негативные примеры полученных корпусов являются семантически более близкими чем негативные примеры других подобных корпусов. Для проверки данного утверждения было произведено сравнение косинусной близости пар предложений негативных примеров существующих наборов данных с сгенерированными наборами данных. Для подсчета косинусной близости предложения векторизировались 5 разными моделями мультиязычной векторизации: MUSE[36] [77], MPnet[37] [71, 72], MiniLM-L12[38] [71, 72], LaBSE[39] [78], LASER[40] [79]. Результаты косинусной близости между парами предложений негативных примеров представлены в Таблице 14.

[36] https://www.tensorflow.org/hub/tutorials/retrieval_with_tf_hub_universal_encoder_qa
[37] https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2
[38] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
[39] https://huggingface.co/sentence-transformers/LaBSE
[40] https://github.com/facebookresearch/LASER/#supported-languages

Таблица 14 – Результаты косинусной близости пар предложений негативных примеров существующих и сгенерированных наборов данных, где векторизация предложений произведена 5 разными моделями. (модель MUSE не поддерживает сербский, чешский, шведский и армянский).

Набор Данных	Языки	MUSE	MPnet	MiniLM	LaBSE	LASER
Сгенерированные на основе MRPC	EN - ES	0.71	0.76	0.74	0.75	0.79
	EN - PT	0.71	0.76	0.74	0.74	0.79
	EN - FR	0.71	0.76	0.74	0.74	0.79
	EN - RU	0.70	0.74	0.73	0.73	0.78
	EN - SR	-	0.74	0.71	0.73	0.78
	EN - CS	-	0.76	0.74	0.74	0.78
	EN - DE	0.70	0.76	0.74	0.74	0.79
	EN - NL	0.71	0.76	0.74	0.74	0.79
	EN - SV	-	0.76	0.74	0.74	0.79
	EN - HY	-	0.73	0.71	0.74	0.80
Neg-1	EN-RU	0.58	0.63	0.62	0.62	0.71
SemEval	EN-ES	0.64	0.66	0.64	0.69	0.74
	EN-AR	0.63	0.65	0.63	0.68	0.73
	EN - NL	0.64	0.67	0.64	0.69	0.74
	EN - IT	0.65	0.66	0.64	0.69	0.73
	EN - DE	0.64	0.66	0.64	0.69	0.73
	EN - FR	0.64	0.66	0.64	0.69	0.74
	EN - TR	0.63	0.67	0.65	0.69	0.74

Учитывая показанные в Таблице 14 результаты, можно утверждать, что получаемы при генерации тестовых наборов данных негативные примеры являются семантически более близкими, чем негативные пары встречающиеся в других существующих наборах данных.

2.2.2.1.2. Параметры обучения

Все выбранные модели были переведены в формат Open Neural Network Exchange[41] (в дальнейшем: ONNX). ONNX - это формат построения и представления моделей глубоких нейронных сетей, который делает модели удобно переносимыми между различными фреймворками, увеличивает скорость их работы в процессе их непосредственного использования, а также уменьшает время требуемое для их обучения. В рамках данной работы был использован ONNX версии 1.12.1.. Модели были переведены в данный формат для более быстрой их работы.

Каждая из моделей была обучена на сгенерированном наборе описанном в *2.2.2.1.1*. В качестве функции потерь была использована модификация функции Cross-Entropy Loss - функция Focal Loss [80].

Во избежании процесса подбора коэффициента скорости обучения, его значение выбиралось с использованием “цикличного коэффициента скорости обучения” [81], который, в отличии от обычного, позволяет менять значение коэффициента для каждого из пакетов данных подаваемых в модель в процессе обучения. При использовании “цикличного коэффициента скорости обучения” задаются верхняя и нижняя границы в пределах которых значение коэффициента будет изменяться. В начале обучения значение коэффициента выставляется на уровне заданной нижней границы, затем начинает увеличиваться исходя из заранее заданной величины шага, после достижения верхней границы значение коэффициента с тем же шагом начинает уменьшаться и так далее. В рамках данной работы, значение нижней граница “цикличного коэффициента скорости обучения” равнялось 0.00001, значение верхней границы было равно 0.0002.

Модели дообучались 3 эпохи, с использование ранней остановки, которая позволяет остановить процесс обучения при незначительной смене значения функции потерь на протяжении нескольких пакетов. Размер пакета данных равнялся 64.

---

[41] https://onnx.ai/

Процесс дообучения и процесс тестирования были произведены на видеокарте (GPU) Nvidia Tesla A100 с видеопамятья в 40 гигабайт, и с использованием процессора (CPU) AMD EPYC 7513 с 32-мя ядрами.

2.2.2.1.3. Эксперименты

В качестве главной цели которая преследовалась в рамках проведения данных экспериментов, является выявление лучшей, с точки зрения точности, “Трансформер”-основанной модели для задачи определения перевода между двумя предложениями. Эксперименты были произведены для данных в которых предложения 10 различных языков вступали в пару только с английскими предложениями, что было сделано исходя из того, что большинство ресурсов содержащихся в интернете, из которых возможно произведение межъязыковых заимствований, написаны на английском языке. В рамках достижения данной цели, каждая из моделей была также обучена при использовании разного количества обучающих данных. Количество обучающих примеров менялось как 100%, 10% и 1% от изначальной обучающей выборки, т.е. 10,000 примеров, 1,000 примеров и 100 примеров. В каждой из неполных обучающих выборок количество пар предложений для различных языков было равно друг другу. Предложения которые входили в неполные выборки выбирались случайным образом.

Второстепенной целью являлось выявление быстродействия каждой из рассматриваемых моделей, для дальнейшего учета данной информации при выборе подходящей модели, для использовании в процессе детального анализа.

Также, одной из целей являлось определение эффективности использования одной общей модели для всех языков или использования по одной модели для каждой языковой группы, или использования по модели для каждого из рассматриваемых языков.

Таким образом, для достижения поставленных целей, отобранные модели были дообучены на каждой из 14 представленных ранее обучающих выборок, с

использованием 100%, 10% и 1% процентов от количества примеров содержащихся в них. Итого, после процесса дообучения количество моделей равнялось количеству отобранных моделей (6 моделей) X количество обучающих выборок (14 выборок) X количество различных величин наборов данных (3 различных величины данных):

$$6 \times 14 \times 3 = 252 \text{ различные модели}$$

Каждая из моделей была протестирована на тестовых наборах язык которых содержался в обучающей выборке конкретной модели, или на тестовых наборах той же языковой группы, что и язык использованный в процессе дообучения. Таким образом, модель дообученная на обучающей выборке содержащей в себе только испанско-английские примеры была протестирована на всех тестовых выборках романской языковой группы (Английский-Испанский, Английский-Португальский, Английский-Французский).

2.2.2.1.4. Результаты

В рамках данного подраздела рассмотрим результаты полученные при использовании 100% обучающих данных, которые представлены на рисунке 2.12. Результаты полученные при использовании 10% и 1% данных представлены и проанализированы в Приложении А.

Возвращаясь к основной цели, на рисунке видно, что почти для каждой тестовой выборки модель XLM-RoBERTa (XLM-R) достигает лучших результатов независимо от обучающих данных на которых она была дообучена. Второй лучший результат в среднем показывает модель mBERT.

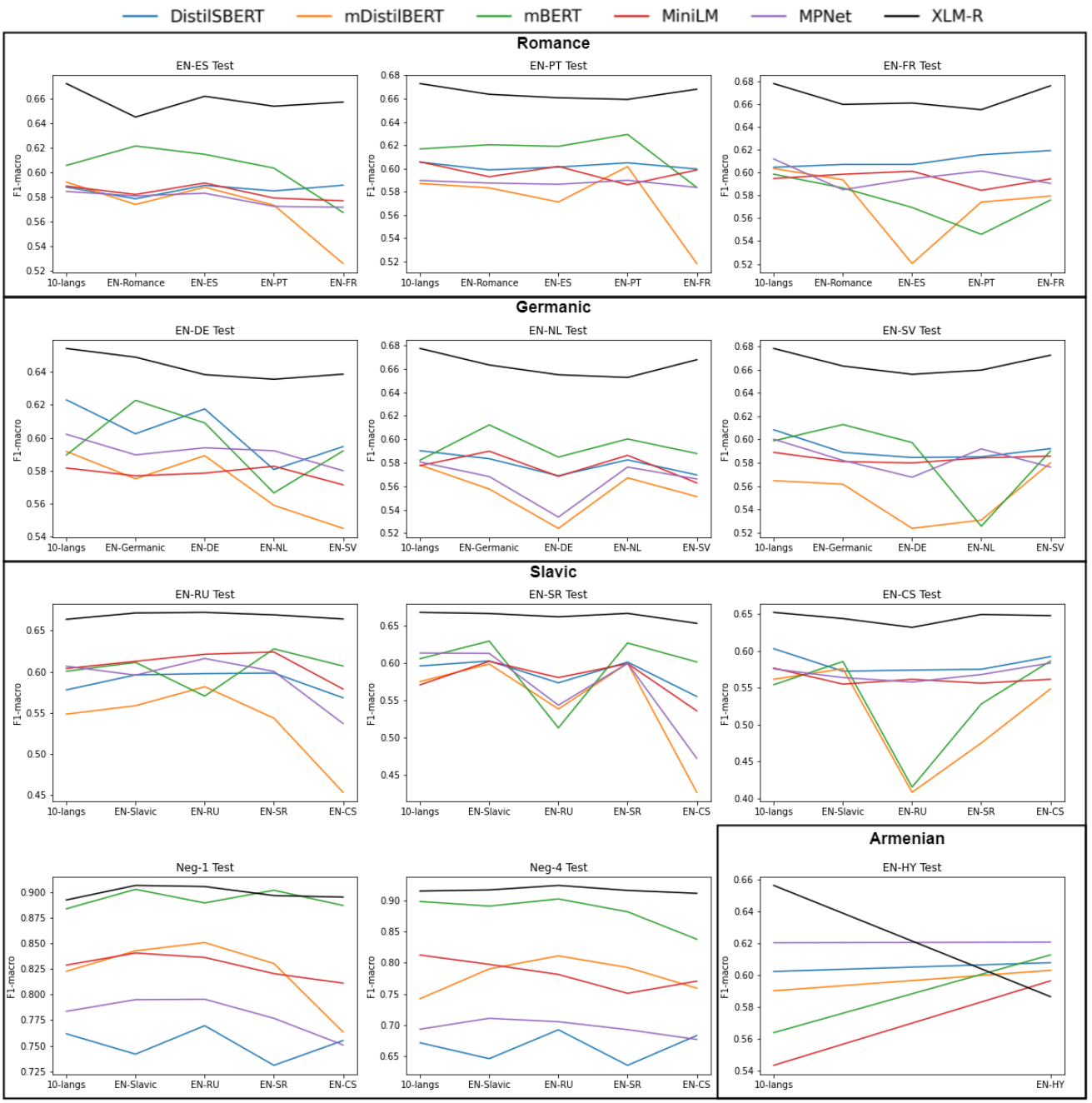

Рисунок 2.12 – Значения оценки F1-macro на каждой тестовой выборке достигнутые дообученными на 100% данных 14 обучающих выборок моделями. Ось X каждого из графиков обозначает обучающую выборку на которой была дообучена модель. Графики разбиты по языковым группам тестовых наборов. Для каждой тестовой выборки результаты показаны только для моделей содержащих в процессе дообучения язык тестовой выборки.

В Таблице 15 отдельно представлено, какая модель для скольких тестовых выборок показывала лучший и второй лучший результаты, а также среднее место

которое занимали модели по всем тестовым выборкам. Исходя из данной статистики, можно сделать вывод, что все остальные модели показали значительно худшие результаты, однако для некоторых тестов модель DistilSBERT выдавала вторые лучшие результаты.

Таблица 15 – Количество тестовых выборок на которых модель достигает лучших (Топ 1) и вторых лучших (Топ 2) результатов, при использовании 100% данных в процессе дообучения. Средний ранг обозначает среднее место которое занимали модели по всем рассматриваемым тестовым выборкам.

Модели	Топ 1	Топ 2	Средний Ранг
mBERT	0/12	8/12	2.6
mDistilBERT	0/12	0/12	4.9
XLM-RoBERTa	12/12	0/12	1.0
MiniLM	0/12	0/12	4.4
MPNet	0/12	1/12	4.2
DistilSBERT	0/12	3/12	3.8

В среднем, результаты для сгенерированных на основе MRPC сложных тестовых выборок, для лучшей модели XLM-RoBERTa варьируются от 59% до 68% по значению оценки F1-macro.

Рассматривая отдельно результаты полученные на более легких тестовых выборках Negative-1 и Negative-4, можно заметить, что результаты сильно отличаются в лучшую сторону. Можно предположить, что на данных выборках модели достигают значительно более высоких результатов из-за схожести тестовых данных с данными на которых были дообучены модели, а также из-за отсутствия сложных отрицательных примеров, которое в наличии у сгенерированных тестовых выборок благодаря корпусу MRPC. Для данных тестовых выборок модель XLM-RoBERTa также показывает лучшие результаты.

Анализируя результаты с точки зрения достижения второй основной цели понять сколько моделей можно использовать для скольких языков, из-за разнообразия полученных результатов для различных моделей невозможно сделать общий вывод для всех моделей. Таким образом, рассмотрим данную проблему только для лучшей модели XLM-RoBERTa.

В рамках романской языковой группы XLM-R достигла лучших результатов дообучаясь на данных содержащих в себе сразу все рассматриваемые языки, обгоняя в среднем на 1-3% модели дообученные на выборках содержащих только конкретные языки или на выборке содержащей все рассматриваемые романские языки. В случае со славянскими языками все модели XLM-R достигли сравнимых результатов отличающихся в пределах одного процента. Для германских языков также, модель дообученная на данных содержащих в себе все языки сразу, показала лучшие результаты. В качестве окончательного аргумента в сторону использования одной модели для всех языков выступили результаты полученные для армянского языка. Для тестовой выборки армянского языка модель дообученная на всех языках сразу достигла значительно лучших результатов (обогнав на 8%) относительно модели дообученной только на английско-армянских примерах.

Таблица 16 – Время затрачиваемое каждой из рассматриваемых моделей на обработку одного примера.

Модели	**Секунда / Пример**
mBERT	0.0039 сек.
mDistilBERT	0.0021 сек.
XLM-RoBERTa	0.0039 сек.
MiniLM	0.0024 сек.
MPNet	0.0039 сек.
DistilSBERT	0.0021 сек.

Переходя к результатам полученным относительно быстродействия каждой из моделей, представленных в Таблице 16, модели XLM-RoBERTa, mBERT и

MPNet работают примерно в 2 раза медленнее чем mDistilBERT, MiniLM и DistilSBERT.

Таблица 17 – Оценка F1-macro для каждой модели обученной на одном из языков романской группы и протестированной на другом языке той же группы с использованием 100% данных во время дообучения.

Модели	**Обучающая выборка**	**Тестовая выборка**		
		EN - ES	EN - PT	EN - FR
mBERT	EN - ES	-	0,619	0,5694
	EN - PT	0,6037	-	0,5459
	EN - FR	0,5677	0,5837	-
mDistilBERT	EN - ES	-	0,5712	0,5204
	EN - PT	0,5736	-	0,5741
	EN - FR	0,5262	0,5183	-
XLM-RoBERTa	EN - ES	-	0,6608	0,6607
	EN - PT	0,654	-	0,6549
	EN - FR	0,6574	0,6682	-
MiniLM	EN - ES	-	0,6018	0,6011
	EN - PT	0,5794	-	0,5844
	EN - FR	0,5772	0,5987	-
MPNET	EN - ES	-	0,5864	0,5945
	EN - PT	0,5726	-	0,6013
	EN - FR	0,5719	0,5837	-
DistilSBERT	EN - ES	-	0,6012	0,607
	EN - PT	0,5852	-	0,6154
	EN - FR	0,5898	0,5995	-

Языковая переносимость моделей. В дополнение к вышеупомянутым результатам, также была посчитана языковая переносимость каждой из

рассматриваемых 6 моделей. То есть, для каждой модели обученной на одном языке какой-то языковой группы были посчитаны результаты данной модели получаемые на тестовых выборках других языков данной языковой группы.

Результаты были посчитаны только для моделей дообученных на 100% тренировочных данных и на сложных тестовых корпусах основанных на MRPC. Результаты для романской группы языков представлены в Таблице 17, результаты других языковых групп описаны в Приложении Б.

В целом, исходя из результатов полученных моделями для всех языковых групп, можно сделать выводы, что в среднем модели обученные на языке определенной подгруппы давали лучшие результаты на тестовых данных языка из той же подгруппы. Однако, тут стоит заметить, что модель показывающая лучшие результаты XLM-RoBERTa на тестовых выборках иберо-романского и западно-германского достигала лучших результатов будучи дообученной на языках других подгрупп: галло-романской и северо-германской соответственно. Таким образом, исходя из представленных результатов можно прийти к выводу, что в случае недостатка данных модель XLM-RoBERTa дообученную на языках какой-то языковой группы можно применять для других языков той же языковой группы.

2.2.2.1.5. Выводы

В данном подразделе рассматривалась задача определения лучшей модели нейронных сетей, которая могла бы быть использована для определения перевода между двумя предложениями на различных языках, для ее дальнейшего применения на этапе детального анализа. В рамках данного раздела было произведено сравнение шести различных моделей с точки зрения их точности, скорости и межъязыковой переносимости.

Исходя из полученных результатов, для дальнейшего использования в задаче детального анализа была выбрана модель XLM-RoBERTa, которая будет обучена

на выборке содержащей в себе все нужные для решения задачи детального анализа языки и будет использоваться для всех языков на которых она была обучена.

2.2.2.2. Обучение итоговой модели детального анализа

После того, как в качестве модели для ее дальнейшего использования на этапе детального анализа была выбрана модель XLM-RoBERTa, было решено дообучить ее на данных имитирующих реальные пары предложений получаемые в процессе работы программы обнаружения межъязыковых заимствований. Исходя из этого в процессе дообучения использовались научные тексты, а отрицательные примеры извлекались с использованием словаря “межъязыковых” синонимов CL-SynDi, описанного в *2.1.3.2*.

2.2.2.2.1. Обучающие данные

Для генерации обучающей выборки, которая бы имитировала данные поступающие в модель детального анализа при реальной работе программы, были использованы предложения из английской Wikipedia, которые имитировали собой данные общей стилистики, а также предложения из различных научных статей различных сфер, взятых случайным образом из Google Scholar, которые имитировали данные написанные в научном стиле.

Позитивные примеры были сгенерированы автоматическим машинным переводом, с помощью инструмента Google Translate, используя для генерации по 4,500 случайных предложений из Wikipedia и научных статей. Перевод этих предложений был произведен таким образом чтобы были получены по 150 позитивных примеров из Wikipedia и научных статей, для всевозможных пар языков (для 6 рассматриваемых языков - 30 всевозможных пар, при учете последовательности предложений на определенных языках). Таким образом, в

итоговой обучающей выборке содержалось 9,000 позитивных примеров, с 300 примерами для каждой пары языков.

Для генерации негативных примеров для каждой пары языков были извлечены по 150 случайных пар предложений из английской Wikipedia и различных научных журналов. Данные предложения были переведены на соответствующие языки для формирования итогового набора негативных примеров. Таким образом, в итоговой обучающей выборке также, как и в случае позитивных примеров, содержалось 9,000 негативных примеров.

Итоговая сгенерированная обучающая выборка содержала в себе 18,000 примеров (Таблица 18).

Таблица 18 – Количество пар предложений в сгенерированной обучающей выборке.

	Wikipedia		**Научные статьи**	
	Положительные	Отрицательные	Положительные	Отрицательные
Обучающая выборка	4500 пар	4500 пар	4500 пар	4500 пар

2.2.2.2.2. Результаты тестирования дообученной модели

В качестве данных для тестирования работы дообученной модели, как и в разделе “Выбор модели” были использованы два набора для пары языков русский-английский - Negative-1 и Negative-4, а также тестовые выборки сгенерированные на основе корпуса перефразирований MRPC. В отличии от этапа выбора модели, тут использовались сгенерированные тестовые выборки только для 5 языков (армянский, русский, испанский, немецкий, французский) рассматриваемых в рамках общей работы алгоритма обнаружения межъязыковых заимствований. Дополнительно, с использованием того же алгоритма, что для генерации обучающей выборки, был сгенерирован тестовый набор состоящий из 2400 примеров (Таблица 19)**,** который также должен был имитировать реальные

данные. Так как данный набор был сгенерирован с использованием того же алгоритма, что для генерации обучающей выборки, следовательно в данном тестовом наборе содержатся примеры сразу для всех рассматриваемых языков, в равном количестве.

Таблица 19 – Количество пар предложений в сгенерированной, по подобию обучающей выборке, тестовой выборке.

	Wikipedia		**Научные статьи**	
	Положительные	Отрицательные	Положительные	Отрицательные
Тестовая выборка	600 пар	600 пар	600 пар	600 пар

Результаты показанные дообученной моделью на всех описанных тестовых выборках представлены в Таблице 20.

Таблица 20 – Результаты полученные дообученной моделью XLM-RoBERTa на различных тестовых наборах для 5 языков.

Набор данных	**Языки**	**Количество пар**	**Полнота**	**Точность**	**F1**
Сгенерированные на основе MRPC	EN - HY	901	0,89	0,74	0,81
	EN - RU	1250	0,91	0,72	0,81
	EN - ES	1457	0,94	0,72	0,82
	EN - FR	1397	0,91	0,72	0,81
	EN - DE	1154	0,94	0.70	0,81
Negative-1	EN - RU	7998	0,89	0,93	0,91
Negative-4	EN - RU	18613	0,84	0,88	0,86
Сгенерированный	6 языков	2400	0,99	0,96	0,97

Дополнительно модель была протестирована на части тестовых данных ACL WMT 2013[42], которые содержат в себе параллельные предложения для различных пар европейских языков и используются для проверки точности систем

[42] http://www.statmt.org/wmt13/

машинного перевода. В качестве тестовых данных были извлечены 3000 примеров для 4 пар языков: английский-русский, английский-французский, английский–немецкий, английский-испанский. Так как тестовая выборка ACL WMT 2013 предназначена для оценки работы систем машинного перевода, в ней не содержатся отрицательные примеры, то есть пары предложений которые не являются переводами друг другами. Исходя из этого на данной выборке была посчитана только метрика точности (accuracy), представленная в Таблице 21.

Таблица 21 – Результаты модели определения перевода на данных тестовой выборки ACL WMT 2013.

Набор данных	**Языки**	**Количество пар**	**Точность**
ACL WMT 2013	EN - RU EN - DE EN - ES EN - FR	3,000	0,92

2.2.3. Использование модели для этапа детального анализа

В процесс детального анализа, после этапа извлечения кандидатов попадали анализируемые предложения и соответствующие им топ-К текстовых фрагментов-кандидатов. Каждый из фрагментов-кандидатов разбивался по предложения, затем все предложения всех фрагментов-кандидатов вступали в пару с анализируемым предложением и подавались в уже дообученную модель бинарной классификации определения перевода между двумя предложениями.

Используемая модель для одного анализируемого предложения может классифицировать несколько предложений-кандидатов в качестве его перевода, что повлияет на точность всей системы. Во избежание данной проблемы, в качестве предложения с которого было произведено заимствование выбиралось то предложение (из списка позитивно классифицированных) в паре с которым анализируемое предложение проходя через модель получала наивысшее значение.

Граница решения модели была выбрана на валидационных данных полученных тем же способом, что и данные на которых она была дообучена. Граница выбиралась исходя из лучшего значения по оценке F0.25 полученного на валидационной выборке. Выбор границы был произведен по оценке F0.25 так как в приоритет ставится точность работы всей системы, а в расчетах оценки F0.25 (2.12) точности дается больший вес.

$$F0.25 = \left(1 + 0.25^2\right) \times \frac{\text{Точность} \times \text{Полнота}}{\left(0.25^2\right) \times \text{Точность} + \text{Полнота}} \quad (2.12)$$

Таким образом, при обработке результатов извлечения кандидатов для одного анализируемого предложения, заимствованием считалась та пара предложений которая проходила выставленную границу, а также получала наивысшую оценку модели. То есть для одного анализируемого предложения в качестве предложения из которого было произведено заимствование могло быть размечено только одного предложение.

Если для какого-то анализируемого предложения в паре с никакими предложениями-кандидатами оценка модели не превышала выставленную границу, то данное анализируемое предложение считалось уникальным.

2.2.4. Искусственные атаки “черного ящика” на языковые модели бинарной классификации

Искусственные атаки на языковые модели все больше обращают на себя внимание, исходя из их потенциального негативного влияния на работу моделей, что подрывает доверие к используемым моделям. В процессе генерации искусственных атак производятся незначительные изменения входных данных, которые вынуждают языковую модель ошибаться [82, 83]. Целью генерации подобных примеров является изменение изначальных входных данных

незаметным для человека образом, при этом добившись ошибки в предсказании модели [84]. Исходя из того, что даже маленькое изменение в текстах может привести к полной смене смысла рассматриваемого текста, сгенерированные примеры искусственных атак должны удовлетворять следующим требованиям:

- изменения должны быть необнаружимы человеком, в частности семантические свойства исходного текста должны быть сохранены;
- звучать естественно (например, с правильными окончаниями слов);
- вынуждать модель ошибаться.

В рамках данной главы рассматривается устойчивость рассмотренных моделей к искусственным атакам, а также представлен новый метод генерации искусственных атак “черного ящика” на языковые модели бинарной классификации, обходящий существующие методы по количеству успешных атак. Метод основан на произведение изменений как на уровне букв, так и на уровне слов.

Для произведения сравнения с другими методами представляемый метод был протестирован на различных существующих тестовых наборов данных на английском языке: IMDB[43], YELP[44] [85], MR[45] [86]. Данные наборы являются тестовыми наборами для задачи анализа тональности.

Все шесть рассматриваемых ранее моделей были протестированы на устойчивость к искусственным атак используя представленный алгоритм атак для задачи бинарной классификации: является ли одно предложение переводом другого с использованием тестового набора NEG-1[46] [42].

[43] https://huggingface.co/datasets/imdb
[44] https://huggingface.co/datasets/yelp_polarity
[45] https://huggingface.co/datasets/rotten_tomatoes
[46] http://nlp.isa.ru/ru-en-text-align-corp/Negative-1/

2.2.4.1. Обзор существующих решений

Алгоритм генерации искусственных примеров для произведения атак на языковые модели состоит из двух основных шагов: выявление “важных” слов в рассматриваемом примере и произведения изменения данных слов.

Существуют два метода генерации искусственных примеров: метод “белого ящика” и метод “черного ящика”. В рамках первого метода при генерации искусственных примеров имеется полная информация о модели: параметры модели, архитектура, и т.д.. Используя данную информацию, можно более эффективно производить атаки на модели [82, 87-89]. Метод “черного ящика” предполагает, что в данном случае атакующий имеет доступ только к выходным значениям модели [90, 91]. В рамках данного раздела рассматривается новый метод “черного ящика”.

Изменения, вносимые в примеры, делятся на два типа: изменения на уровне слов и изменения на уровне букв. Атаки на уровне слов заменяют, удаляют или вставляют целые слова в рассматриваемых примерах. Некоторые методы производят изменения слов с использование фиксированных правил, на основе информации о части речи слова или его именованной сущности [92, 93]. Однако данные методы не гарантируют семантическую идентичность примеров и их естественное звучание. Замена слов в различных работах производилась на основе словарей синонимов [91, 93] или с использование векторных представлений слов и близости данных представлений, дополнительно фильтруя стоп-слова и антонимичные слова [95]. Также используются методы замены слов на основе предсказаний языковых моделей, позволяющие сохранять семантическую близость искусственного и оригинального примеров [90, 96]. Данные методы показывают улучшенный процент атак, при этом лучше сохраняя семантическую близость искусственных примеров к их оригиналам.

Атаки на уровне букв основаны на различных действиях, связанных с буквами: удаление, добавление, рокировка, замена и т.д. [87, 97, 98]. Так, например, метод представленный в [89] основывается на использовании опечаток,

исходя из расположения букв на клавиатуре, т.к. такие опечатки выглядят естественно и являются незаметными [99], в то же время вынуждая модель ошибаться.

После нахождения кандидатов для смены слов в тексте, следующей задачей является выбор лучших из кандидатов влияющих на модель в большей степени, чем остальные. Для определения лучших кандидатов могут быть использованы жадные алгоритмы [93, 100], которые, например, используют лучевой поиск для выбора слов влияющих на модель в большей степени.

2.2.4.2. Генерация искусственных примеров на уровне букв

В данном подразделе представляется новый метод генерации искусственных атак на уровне букв на языковые модели использующие WordPiece токенизацию [101]. С использованием данной токенизации выбирались варианты для изменения слов.

2.2.4.2.1. Определение порядка слов для произведения изменений

На первом шаге слова в тексте сортируются по “важности”, для произведения дальнейших изменений именно в этом порядке. “Важность” слов считалась также, как и в множестве существующих методов [93, 95]. Допустим, имеется:

- текст $T = \{w_1, w_2, \ldots, w_n\}$, где w_i i-ое слово в тексте;
- $T\backslash w_i = T\backslash\{w_i\} = \{w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n\}$ текст получаемый при удалении слова w_i;
- Значение предсказание модели M для некоторой метки y - $M_y(-)$;

Значение важности слова I_{w_i} считается как разница между значениями предсказания модели для метки y до и после удаления слова w_i:

$$I_{w_i} = M_y(T) - M_y(T \backslash w_i) \tag{2.13}$$

I_{w_i} считается для каждого слова в тексте и, исходя из данных значений, слова сортируются. Дальнейшие изменения в словах происходят в убывающем порядке, начиная со слов с наибольшим значением важности.

2.2.4.2.2. Варианты действий с буквами

После получения последовательности, по которой будут производится изменения, рассмотрим варианты действий для произведения этих изменений. В качестве действий использовались представленные в [89], с дополнительным ограничением на использование цифр и гомоглифов:

- Удаление - удаление случайной буквы
- Вставка - рядом со случайной буквой в слове вставка буквы, которая: а) находится вокруг случайной буквы на клавиатуре; б) та же буква, что и случайная. Дополнительно, если вставка производилась справа от последней буквы слова, то в качестве кандидатов на вставку также входили буквы находящиеся рядом с кнопкой “Пробел”.
- Замена - замена случайной буквы слова на букву находящуюся на раскладке клавиатуры вокруг заменяемой.
- Рокировка - рокировка двух соседних букв слов.

Все рассмотренные действия имитируют ошибки, возникающие при быстром печатании текстов. При использовании данных действий, одно слово подвергалось только одному действию.

2.2.4.2.3. Генерация искусственных примеров на основе WordPiece токенизации

Основной мотивацией использования информации предоставляемой WordPiece токенизацией была максимизация вероятности атаки без использования жадного алгоритма нахождения худшего кандидата для модели. Таким образом, используя действия представленные в *2.2.4.2.2.*, получались все возможные кандидаты на изменение исходных слов текста. Каждое оригинальное слово, и его кандидаты, подвергались процессу WordPiece токенизации, после чего кандидаты отфильтровывались способом “минимального пересечения токенов”.

Минимальное пересечение токенов (МПТ) - считается пересечение между токенами оригинального слова и токенами всех кандидатов. В качестве кандидатов оставляются только те, кто имеет минимальное значение пересечения. Например, для слова *“melodrama” → [’mel’, ’##od’, ’##rama’]* будет оставлен кандидат *“melodarma” → [’mel’, ’##oda’, ’##rma’]*, вместо *“melodramas” → [’mel’, ’##od’, ’##rama’, ’##s’]*.

Идея подобной фильтрации состоит в том, что большая разница между токенами оригинального слова и токенами его кандидатов приведет к большей семантической разнице между оригинальным и искусственным примером.

После фильтрации кандидатов получаем финальный список кандидатов с изменениями на уровне букв для его дальнейшего использования в процессе генерации искусственных примеров.

2.2.4.3. Генерация искусственных примеров на уровне слов

Для получения семантически корректных изменений различные методы используют словари синонимов (WordNet, BabelNet, HowNet [102], и т.д.) или близость по векторным представлениям слов. Однако, данные подходы имеют свои проблемы. При использовании словарей синонимов может оказаться, что

некоторые “важные” слова не присутствуют в словаре, что повлияет на успешность атак. Основываясь на близости векторных представлений слов, можно получить кандидаты семантически далекие от оригинального слова.

Во избежании данных проблем было предложено использовать синонимы получаемые при помощи модели ChatGPT[47]. Использование ChatGPT открывает возможности к получению синонимов практически для любого слова, в том числе и для слов отсутствующих в существующих словарях синонимов. Например, для слова “oooooh”, которое с большой вероятностью не присутствует ни в одном из существующих словарей, ChatGPT сгенерировало такие синонимы, как: “ohh”, “ahhh”, “ooh”, “woo”. Также модель способна генерировать синонимы для собственных имен, например, для слова “Marianne” были сгенерированы синонимы “Mary”, “Maria”, “Marie”.

```
generate multiple synonyms for the following words(set the
synonyms on the same line with the corresponding word):
   1. he
   2. threw(verb)
   3. into
   4. the
   5. wastebasket(noun)
   6. letter(noun)
```

Рисунок 2.13 – Пример запроса подаваемого в модель ChatGPT для генерации синонимов.

Для получения синонимов с использованием модели ChatGPT, были сгенерированы запросы подаваемые модели, с помощью которых было бы возможно получить несколько синонимов для каждого из слов текста. При генерации запроса, для получения корректных синонимов, дополнительно учитывались части речи слов, которые в свою очередь было получены с помощью библиотеки Stanza. Для слов, которые являлись глаголами, существительными,

[47] https://openai.com/blog/chatgpt

прилагательными, наречиями, именами собственными и междометиями, метка их части речи добавлялась рядом с этими словами в запросе. Финальный запрос для текстов английского языка показан на рисунке 2.13.

2.2.4.3.1. Стратегии генерации искусственных примеров на основе синонимов ChatGPT

При замене слов мы хотим, чтоб данные замены были максимально естественными, сохраняя свои морфологические признаки (род, падеж, число, и т.д.). Для достижения данной цели мы произвели сравнение трех разных методов генерации кандидатов.

Морфологический анализ и словоизменение (MAC). В рамках данного метода производится имитация использования обычного словаря синонимов. При получении синонимов с помощью ChatGPT запрос подается для лемматизированных оригинальных слов, тем самым в ответ генерируются также лемматизированные синонимы. После чего для оригинального слова извлекаются его морфологические признаки, с использованием которых производится процесс словоизменения полученных синонимов. Процессы извлечения морфологических признаков и словоизменений производились с использованием библиотек spaCy[48] [103] и pymorphy2[49] [104] для текстов английского и русского языков соответственно. Далее словоизмененные синонимы использовались в качестве кандидатов на смену оригинальных слов.

BERT-основанное предсказывание маскированных токенов (BERT MT). Данный метод сохранения морфологических признаков для синонимов основан на использовании BERT-основанных моделей для предсказания маскированных токенов. В данном случае с помощью ChatGPT также получаются лемматизированные синонимы, после чего производятся попытки предсказать их окончания, тем самым ставя их в правильную морфологическую форму.

[48] https://spacy.io/
[49] https://github.com/pymorphy2/pymorphy2

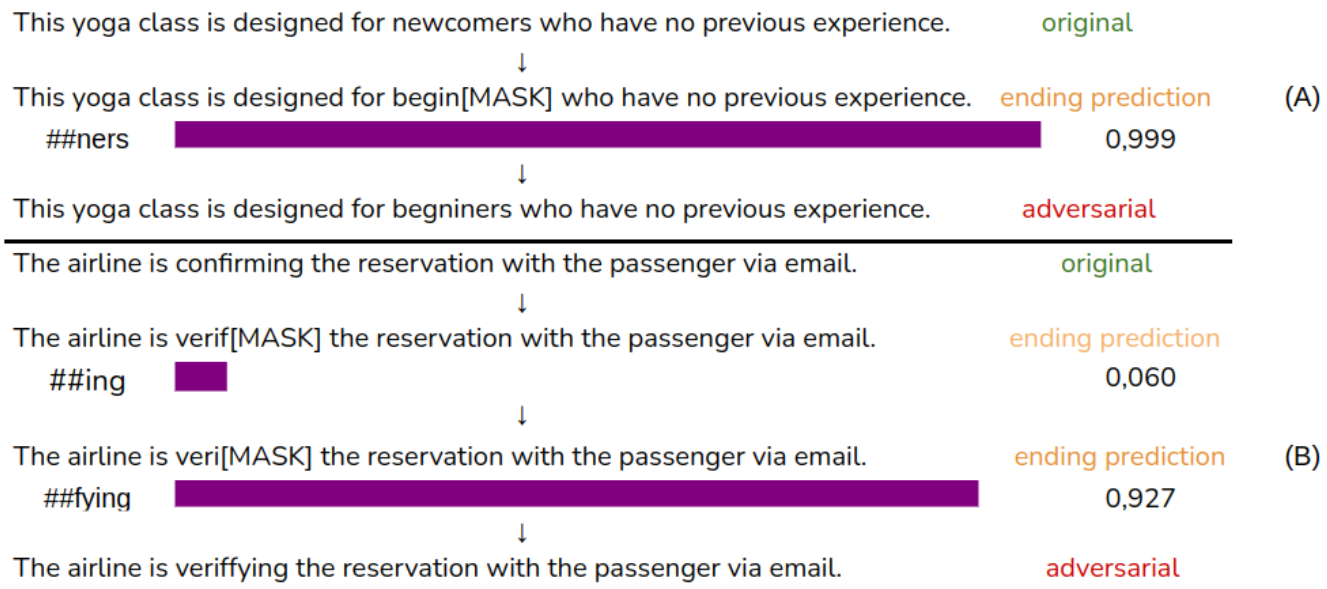

Рисунок 2.14 – Пример работы BERT-основанного предсказания маскированных токенов для получения кандидатов-синонимов с правильными морфологическими признаками.

Предсказание окончаний производилось в следующие 3 шага:

- Оригинальное слово заменяется его лемматизированным синонимом-кандидатом;
- Синоним-кандидат токенизируется с помощью WordPiece токенизатора;
- Исходя из результата токенизации, процесс предсказания производится двумя способами:
    - Если синоним-кандидат токенизируется на больше чем один токен, то его последний токен маскируется, после чего производится его предсказание;
    - Если синоним-кандидат токенизируется как цельный токен, процесс предсказания производится итеративно. На первом шаге токен маскировки “[MASK]” добавляется в конце слова, после чего производится предсказывание. Далее, начиная с второго шага последние буквы кандидата удалялись одна за одной, заменяясь токеном маскировки. На 5 итерации процесс останавливался. Имея вероятности возвращенные BERT-основанной моделью на каждом из

шагов для предсказаний маскированных токенов в качестве итогового окончания выбиралось окончание с самой высокой вероятностью.

Данный процесс показан на рисунке 2.14.

ChatGPT сохранение морфологических признаков, морфологический анализ и словоизменение (СМП МАС). Модифицировав запрос, подаваемый ChatGPT для каждого оригинального слова, возвращались синонимы с сохранением морфологических признаков. Пример модифицированного запроса для текстов русского языка показан на рисунке 2.15.

Тем не менее, для некоторых оригинальных слов все же возвращались синонимы в их лемматизированной форме. Такие синонимы были дополнительно словоизменены тем же способом, что был представлен в методе **МАС**.

```
Provide synonyms for the following words in Russian, keeping
the same grammatical form (e.g. case, gender, number). Set the
synonyms on the same line with the corresponding word. For
example, if the word is 'города', the answer could be 'города
- местности, поселения, мегаполиса'.
  1. солдаты(noun)
  2. грелись(verb)
  3. около(adverb)
  4. костра(noun)
```

Рисунок 2.15 – Пример запроса в модель ChatGPT, позволяющего сохранить морфологические признаки оригинального слова на этапе генерации синонимов.

С использованием одного из вышеописанных методов получается список из кандидатов с изменениями на уровне слов для его дальнейшего использования в процессе генерации искусственных примеров.

После получения списка кандидатов с изменениями на уровне слов одним из вышеописанных методов, для каждого из синонимов-кандидатов генерируется их версия с опечаткой тем же методом что представлен в *2.2.4.2*. После чего, объединяя список кандидатов без и с опечатками, получаем окончательный список кандидатов с изменениями на уровне слов.

2.2.4.4. Итоговый метод генерации искусственных примеров

Получив для оригинального слова отдельно множество кандидатов с изменением на уровне букв и множество кандидатов с изменением на уровне слов, данные множества объединяются, формируя множество кандидатов для итогового алгоритма генерации искусственных примеров.

На первом шаге алгоритм берет самое “важное” слово и заменяет его кандидатами из полученного множества кандидатов. При генерации искусственных примеров на каждом шаге в итоге выбирается кандидат, который больше всего влияет на предсказание языковой модели. Если, после замены одного слова, модель не начинает ошибаться, алгоритм переходит к следующему по важности слову. Таким образом, изменению подвергаются до 40% слов. Если модель в определенный момент начинает ошибаться, то алгоритм останавливается, и полученный искусственный пример считается примером успешной атаки. Для тех примеров, при изменении которых на больше чем 40%, модель все еще не ошибалась, атака считалась неуспешной.

Для успешных атак, после процесса изменений оригинальных примеров ,производилась операция минимизации изменений, для получения еще более незаметных для человека и естественных искусственных примеров.

Для минимизации изменений был подсчитан “урон” выдаваемый модели каждым из заменяемых слов на каждом шаге. После чего, слова были отсортированы в возрастающем порядке исходя из значений “урона”. Начиная с слова нанесшего меньше всего “урона”, изменения аннулировались, возвращая оригинальное слово, после чего заново производилась попытка атаки. Если после возвращения оригинального слова атака все еще оставалась успешной, то оригинальное слово оставлялось тем самым минимизируя изменения, после чего алгоритм переходил к следующему по “урону” слову. Если же атака становилась безуспешной, изменение возвращалось и алгоритм переходил к следующему по “урону” слову. Доходя до конца отсортированного списка, алгоритм шел в обратном порядке и так до тех пор, пока список не подвергался изменению, т.е.

невозможно было вернуть никакое оригинальное слово не испортив успешность атаки. Псевдокод минимизации изменений представлен в Алгоритме 1.

Алгоритм: Минимизация Изменений
Вход: *оригинальныйТекст*, *атакованныйТекст*, *изменения*
Выход: *атакованныйТекст* с минимальным количеством изменений
оригинальныйТекст: оригинальный текст из набора данных
атакованныйТекст: измененный текст вынуждающий модель ошибаться
изменения: информация о всех изменениях (слово и его изменение)

```
1: procedure МинимизироватьИзменения
2:     урон = [ ]
3:     временныйТекст = оригинальныйТекст
4:     for each слово in изменения do
5:         оценка = предсказание(временныйТекст)
6:         временныйТекст = заменить(временныйТекст, слово)
7:         урон ← |предсказание(временныйТекст) - оценка|
8:     Отсортировать изменения по урону
9:     новыеИзменения = [ ]
10:    while изменения ≠ новыеИзменения do
11:        новыеИзменения = изменения
12:        изменения = перевернуть(изменения)
13:        for each слово in изменения do
14:            временныйТекст = вернутьОригинальноеСлово(атакованныйТекст,
    слово)
15:            if успешнаяАтака(временныйТекст) then
16:                атакованныйТекст = временныйТекст
17:                удалить слово из изменения
18:        return атакованныйТекст
```

Алгоритм 1: Псевдокод процесса минимизации изменений после совершения успешной атаки.

2.2.4.5. Результаты

Все представляемые в дальнейшем эксперименты и сравнения различных алгоритмов были произведены на 1000 примерах для каждого из английских наборов данных. Для каждого из методов генерации искусственных атак были подсчитаны следующие метрики:

- Процент успешных атак (AR) - отношение количества успешных атак к количеству всех примеров.
- Процент изменения (PR) - средний процент измененных слов в примерах
- Дистанция Левенштейна (Lev.) - дистанция Левенштейна на уровне символов между искусственным примером и его оригиналом.
- Семантическая близость - значение семантической близости между искусственным примером и его оригиналом подсчитанное с помощью Universal Sentence Encoder[50] (USE) [105] в случае с английскими наборами данных и с помощью Multilingual Universal Sentence Encoder (MUSE) [77] в случае с русскими наборами данных.

Отдельно представляются результаты полученные с использованием только изменений на уровне букв, только изменений на уровне слов и с использованием всех типов изменений.

Таблица 22 – Средние значения метрик по наборам данных IMDB, MR, YELP для представляемого метода изменений на уровне букв и существующего метода DeepWordBug .

	AR	**PR**	**Lev.**	**Вставка**	**Удаление**	**Замена**	**Рокировка**
МПТ	**88.2%**	**6.5%**	**5.53**	47.2%	13.9%	31.8%	7.1%
DeepWordBug	71.6%	12.8%	8.41	-	-	-	-

Изменения на уровне букв. Представляемый метод изменения на уровне букв сравнивается с лучшим методом генерации искусственных примеров с использование только изменений на уровне букв DeepWordBug [98]. Исходя из результатов представленных в Таблице 22, представляемый метод значительно превзошел DeepWordBug по всем метрикам. В данном случае не была подсчитана семантическая близость искусственных примеров с их оригиналами, т.к. опечатки считались естественными, тем самым не влияя на семантическую близость.

[50] https://www.tensorflow.org/hub/tutorials/semantic_similarity_with_tf_hub_universal_encoder

Дополнительно, в таблице показан процент каждого из действий применимого при получении примеров успешной атаки.

Изменения на уровне слов. Все три представляемых метода были протестированы на трех английских наборах данных. Средние показатели всех метрик для всех трех подходов представлены в Таблице 23. Также, в таблице указан процент использования просто синонимов и синонимов с опечатками в процессе генерации искусственных примеров. Исходя из результатов данные методы достигают лучших показателей процента успешных атак и процента изменений чем многие существующие алгоритмы (Таблица 25). В основном слова заменяются на синонимы с опечатками, что объясняется еще большим отдалением от оригинального предложения после опечатки.

Таблица 23 – Средние значения метрик по наборам данных IMDB, MR, YELP для представляемых методов изменений на уровне букв (%syns и %syns* обозначают процент использования просто синонимов и синонимов с опечатками соответственно)

	AR	**PR**	**USE**	**Lev.**	**%syns**	**%syns***
BERT MT	92.87%	6.11%	**0.95**	28.37	11.89%	88.11%
MAC	**93.95%**	**5.98%**	**0.95**	**26.33**	8.92%	91.08%
СМП MAC	92.35%	6.51%	0.94	34.82	9.87%	90.13%

Изменения на уровне и слов и букв. Результаты использования итогового метода представленного в *2.2.4.4* представлены в Таблице 24 по отдельности для каждого из методов изменения на уровне слов и каждого тестового набора данных.

Также как и в случае с изменениями только на уровне слов, больше всего оригинальные слова заменяются на синонимы с опечатками, однако стоит также заметить, что существенное количество оригинальных слов было изменено только на уровне букв, что приводит к значительному снижению метрики дистанции Левенштейна и делает измененный текст более незаметным. Также исходя из

результатов видно, что в среднем лучших результатов добиваются методы **MAC** и **BERT MT**.

Таблица 24 – Метрики полученные на каждом из наборов данных всеми представляемыми методами генерации искусственных примеров.

		AR	PR	USE	Lev.	%syns	%syns*	%typos
MR	**BERT MT**	**93.10%**	9.99%	0.89	7.18	7.33%	44.98%	47.69%
	MAC	92.76%	**9.93%**	**0.90**	**7.03**	4.27%	49.61%	46.12%
	CMП MAC	**93.10%**	10.15%	0.89	7.65	5.28%	45.31%	49.41%
IMDB	**BERT MT**	**98.80%**	2.77%	**0.98**	24.38	8.75%	52.80%	38.45%
	MAC	**98.80%**	**2.60%**	**0.98**	**23.90**	6.43%	59.86%	33.71%
	CMП MAC	97.90%	2.70%	**0.98**	24.43	5.75%	53.56%	40.70%
YELP	**BERT MT**	96.92%	5.15%	**0.96**	24.38	6.53%	56.40%	37.07%
	MAC	**97.02%**	**4.96%**	**0.96**	**24.06**	6.73%	61.73%	31.54%
	CMП MAC	96.61%	5.25%	**0.96**	25.07	4.88%	54.57%	40.56%

Таблица 25 – Средние значения метрик по наборам данных IMDB, MR, YELP для представляемых методов и лучших существующих методов.

	BERT MT	MAC	PWWS	TextFooler	Tampers	Bert-attack	TextBugger
AR	**96.27%**	96.19%	91.31%	94.56%	95.35%	83.06%	78.62%
PR	5.97%	5.83%	8.79%	12.65%	**4.43%**	10.56%	22.44%
USE	**0.95**	**0.95**	0.92	0.90	0.93	0.88	0.93
Lev.	18.65	**18.33**	33.93	52.71	35.25	118.29	35.48

В Таблице 25 произведено сравнение двух лучших представляемых методов (PWWS [93], TextFooler [100], Tampers [94], Bert-attack [99], TextBugger [106]) с лучшими существующими методами. Результаты сравнения усреднены по трем английским наборам данных. Исходя из результатов можно утверждать, что представляемые методы достигают лучших значений процента успешных атак,

при этом будучи более близкими семантически и более близкими по дистанции Левенштейна.

Также было произведено сравнение временной сложности быстрейшего из представляемых методов с предыдущим лучшим методом генерации искусственных атак **Tampers**. Сравнение производилось на одной видеокарте Nvidia RTX 3090. По итогу представляемый метод для различных наборов данных показал на от 30% до 65% быстрее результат (Таблица 26).

Таблица 26 – Сравнение временной сложности генерации одного искусственного примера методами MAC и Tampers.

	MR	**IMDB**	**YELP**
Tampers	9.5 сек./пример	98.5 сек./пример	79,6 сек./пример
MAC	**6.8 сек./пример**	**41.0 сек./пример**	**27.7 сек./пример**

2.2.4.6. Устойчивость моделей к искусственным атакам

Все шесть моделей рассмотренных ранее, были подвергнуты искусственным атакам с использованием MAC алгоритма генерации атак. Устойчивость моделей измерялась с использованием тестового набора NEG-1, для задачи бинарной классификации: является ли одно предложение переводом другого. В рамках произведения атак, изменениям подвергались только предложения на русском язык. Результаты атак для каждой из рассматриваемых моделей представлены в Таблице 27.

По результатам видно, что некоторые модели более уязвимые, а некоторые менее уязвимые к искусственным атакам. Дополнительно, результаты доказывают, что представляемый новый метод генерации атак является переносимым и на другие языки.

Таблица 27 – Значение метрик атакуемых моделей для русско-английского набора Neg-1 (изменения производились над предложением на русском языке).

Model	AR	PR	USE	Lev.
mBERT	90,92%	14,29%	0,91	12,09
mDistilBERT	74,37%	16,10%	0,90	13,59
XLM-RoBERTa	93,89%	13,07%	0,93	13,30
MiniLM	65,64%	14,17%	0,92	13,20
MPNet	70,08%	14,30%	0,92	12,74
DistilSBERT	81,60%	12,30%	0,93	9,64

2.2.4.7. Выводы

В рамках данного подраздела был представлен новый метод генерации искусственных примеров для произведения атак на языковые модели, который обошел все существующие методы по проценту успешности атак, семантической близости сгенерированных примеров с их оригиналами и по дистанции Левенштейна между ними. Метод также был опробован на задаче бинарной классификации обнаружения является ли одно предложение переводом другого, изменяя оригинальные примеры на русском языке, в рамках оценки устойчивости различных моделей к новому алгоритму генерации атак. Результаты изменений примеров русского языка, также, доказали переносимости представляемого нового метода на другие языки.

2.2.5. Методика выбора модели для этапа детального анализа

С учетом результатов полученных при оценки эффективности различных моделей на сгенерированных тестовых наборах данных (Глава 2.2.2.) и результатов проверки моделей на устойчивость к новому методу генерации

искусственных атак (Глава 2.2.4.) была разработана методика выбора модели для ее использования в рамках этапа детального анализа.

Методика содержит в себе 4 этапа:

1. Генерация сложных тестовых выборок.
2. Проверка качества моделей на сгенерированных выборках.
3. Проверка моделей на устойчивость к новому методу генерации искусственных атак.
4. Выбор модели в зависимости от степени угрозы быть подверженным искусственным атакам.

В рамках дальнейшей работы предполагается, что угроза быть подверженным искусственным атакам минимальным атакам и в качестве модели детального анализа используется XLM-RoBERTa.

2.2.6. Выводы

В этом разделе был рассмотрен этап детального анализа, основанный на попарном сравнении текстовых фрагментов полученных после этапа извлечения кандидатов. Было произведено сравнение различных больших языковых моделей, для выбора лучшей модели для ее дальнейшего применения в решении поставленной задачи детального анализа. Были изучены наборы данных, которые могли быть применимы в процессе дообучения моделей и их тестирования. В рамках данной главы также был предложен процесс генерации сложных тестовых наборов определения перевода между двумя предложениями, основанных на корпусе перефразирования MRPC. С использованием предложенного метода были сгенерированы сложные тестовые наборы определения перевода для 10 языков в паре с английским.

Среди рассмотренных моделей лучшие результаты показала модель XLM-RoBERTa, которая и была выбрана для дальнейшей работы на этапе детального анализа. Для дальнейшего использования выбранной модели была

также сгенерирована обучающая выборка имитирующая реальные данные получаемые во время работы всей системы обнаружения межъязыковых заимствований. В итоге была получена модель бинарной классификации обнаружения перевода между двумя предложениями, которая и использовалась в дальнейшем в работе всей системы.

Также, в рамках данного подраздела был представлен новый метод генерации примеров искусственных атак, обходящий по метрике доли успешных атак все существующие методы. Используя новый метод генерации искусственных атак различные модели были протестированы на устойчивость к искусственным атакам.

На основании результатов полученных при оценке эффективности различных моделей на сгенерированных тестовых наборах данных и результатов проверки моделей на устойчивость к новому методу генерации искусственных атак была разработана методика выбора модели для ее использования в рамках этапа детального анализа.

Глава 3. Сравнительный анализ методов обнаружения межъязыковых заимствований

В данной главе рассматриваются различные тестовые корпусы обнаружения межъязыковых заимствований для исследуемых языков. Производится оценка работы представляемого алгоритма и сравнение его точности работы с точностью других подобных алгоритмов.

В первом разделе описываются методы построения различных существующих корпусов. В дополнение к существующим корпусам описывается, также, ново-сгенерированный корпус для всех рассматриваемых 5 языков в паре с английским. Во втором разделе производится сравнение результатов полученных на данных корпусах между представляемым алгоритмом и SOTA алгоритмами для каждого из корпусов.

3.1. Метрики оценки качества обнаружения заимствований

Метрики оценки качества обнаружения заимствований, используемые в рамках различных статей, были представлены в [107]. В рамках данной статьи описаны два типа метрик точности и полноты для оценки качества обнаружения заимствований: micro и macro. Также, описана метрика *Granularity,* отвечающая за цельность нахождения конкретного заимствованного фрагмента. Все описанные метрики объединяются в единую формулу и образуют новую метрику *plagdet.*

Микро-, макро- точность, полнота. Проверка точности и полноты происходит на уровне символов, входящих в искомые, и обнаруженные фрагменты текста.

В наличии множество $S = \{s_1, ..., s_n\}$ всех фрагментов, из которых были произведены заимствования, где $\{s_1, ..., s_n\}$ в свою очередь являются множествами последовательных букв, из которых состоит некоторый фрагмент некоторого документа $D_{\text{источник}}$, $s = \{(i, D_{\text{источник}}), ..., (i + n, D_{\text{источник}})\}$, где $i, ..., i + n$ это индексы символов в конкретном документе-источнике $D_{\text{источник}}$. Аналогично, имеется подобное множеству S множество $R = \{r_1, ..., r_n\}$, всех найденных алгоритмом фрагментов, которые также содержат в себе информацию о индексах символов документов.

Дополнительно, перед подсчетом метрик, ставится условие, что фрагменты, из которых были произведены заимствования, не могут иметь пересечений в символах между собой $\{s_1, ..., s_n\}$. В случае с найденными фрагментами $\{r_1, ..., r_n\}$ такое условие не выдвигается.

Имея вышеописанные обозначения, определим сначала метрики микро-точности (3.1) и микро-полноты (3.2):

$$Precision_{micro}(S,\ R) = \frac{\left| \bigcup\limits_{(s,r) \in (S,R)} (s \sqcap r) \right|}{\left| \bigcup\limits_{r \in R} r \right|}, \quad (3.1)$$

$$Recall_{micro}(S,\ R) = \frac{\left| \bigcup\limits_{(s,r) \in (S,R)} (s \sqcap r) \right|}{\left| \bigcup\limits_{s \in S} s \right|}, \quad (3.2)$$

где $s \sqcap r = \begin{cases} s \cap r \\ \varnothing \end{cases}$

Макро-точность (3.3) и макро-полнота (3.4) определяются следующим образом:

$$Precision_{macro}(S,\ R) = \frac{1}{|R|} \sum_{r \in R} \frac{\left| \bigcup\limits_{s \in S} (s \sqcap r) \right|}{|\ r\ |}, \quad (3.3)$$

$$Recall_{macro}(S,\ R) = \frac{1}{|S|} \sum_{s \in S} \frac{\left| \bigcup\limits_{r \in R} (s \sqcap r) \right|}{|\ s\ |}. \quad (3.4)$$

Granularity. Данная метрика нацелена на определение другого аспекта успешного нахождения заимствований, а именно, нахождения одновременно цельных кусков заимствований. То есть, для некоторого фрагмента s из которого было произведено заимствование, проверить найден ли он был как цельный фрагмент или по разным его частям. В идеале каждый из таких фрагментов должен быть найден как цельный. Для оценки данного аспекта, в рамках [107], представляется метрика Granularity (3.5), определенная следующим образом:

$$Granularity(S,\ R)\ =\ \frac{1}{\left|S_R\right|}\sum_{s\ \in\ S_R}\left|R_s\right|, \tag{3.5}$$

где $S_R \subseteq S$ - это множество тех фрагментов, с которых было произведено заимствование, которые были частично или полностью найдены в R. $R_s \subseteq R$ - это множество всех найденных подфрагментов для конкретного фрагметна s, из которого было произведено заимствование.

Plagdet. В данной метрике авторы [107] объединили в себе представленные выше метрики с использование гармонического среднего микро или макро точности и полноты с некоторым коэффициентом α. Данная метрика (3.6) выглядит следующим образом:

$$Plagdet(S,R)\ =\ \frac{F_\alpha}{log_2(1+Granularity(S,R))}. \tag{3.6}$$

3.2. Тестовые корпусы обнаружения межъязыковых заимствований

Для пар языков английский-русский, английский-французский, английский-испанский уже существовали тестовые корпусы обнаружения межъязыковых заимствований. Для этих пар языков было произведено сравнение представляемого алгоритма с результатами, которые получают SOTA алгоритмы на данных тестовых корпусах. В дополнение к этому, так как не существовало подобных корпусов для пары языков английский-армянский, были сгенерированы 5 новых корпусов для всех рассматриваемых пар языков. Результаты алгоритма представлены и для этих ново-сгенерированных корпусов.

3.2.1. Корпус CrossLang

Корпус CrossLang [13] – это автоматически сгенерированный тестовый корпус обнаружения межъязыковых заимствований между документами русского и английского языков. Перед созданием данного корпуса авторы произвели исследование: из скольких ресурсов в среднем производятся заимствования в рамках одного анализируемого документа во время моноязычных заимствований. В рамках исследования выяснилось, что в одном документе, в котором существовали заимствования, они производились из максимум до 10 различных документов.

В рамках данного корпуса, исходя из того, что при совершении моноязычных заимствований, часто бывают ситуации, в которых производится полное копирование всего текста (т.е. бывает очень высокий процент заимствования), анализируемые документа содержали до 80% заимствований. Нижней же границей была выбрана доля заимствований в 20%.

При генерации корпуса, в качестве документов проверочной базы авторы использовали около 100,000 статей из английской Wikipedia. Анализируемые документы были извлечены случайным образом из русской Wikipedia в количестве 316 статей. Процесс генерации производился в 3 этапа. На первом этапе анализируемые документы переводились, с использованием машинного перевода, на английский язык и для каждого анализируемого документа с помощью функции TF-IDF находились ближайшие к нему 500 документов из заранее отобранных 100,000. Далее, вторым шагом выбирались от 1 до 10 документов из этих 500, и случайные предложения этих документов переводились на русский. Третьим шагом, переведенные предложения заменяли оригинальные предложения анализируемых документов. Заменялось такое количество предложений, которое являлось бы от 20% до 80% всего анализируемого документа. Процент замены предложений для каждого анализируемого документа отбирался случайным образом.

3.2.2. Параллельные корпусы

Для проверки работы алгоритма на паре языков английский-французский существуют несколько параллельных корпусов представленных в [108] , которые являются подкорпусами различных параллельных корпусов, которые используются для различных мультиязычных задач. В рамках сравнения алгоритмов использовались следующие 3 корпуса:

- JRC-Acquis[51]
- Amazon Product Review (APR) [109]
- Conference Papers (TALN) [110]

По очереди рассмотрим каждый из них.

JRC-Acquis. Это корпус параллельных документов, содержащих в себе различные законодательные тексты Евросоюза, переведенные на все 23 официальных языка используемых странами входящими в его состав. Однако, в рамках подкорпусов представляемых в [108] использовались только документы английского, французского и испанского языков.

Amazon Product Review (APR). Корпус основан на отзывах о продуктах представленных на сайте Amazon[52] для 4 языков: немецкий, французский, японский, английский. Корпус был изначально создан для задачи межъязыкового анализа тональности текстов. Параллельные предложения для отзывов были получены с использованием Google Translate. В рамках представляемого подкорпуса, также, была использована только часть отзывов на французском языке.

Conference Papers (TALN). Данный корпус содержит в себе тексты научных статей, которые были опубликованы сначала на одном языке, а потом были переведены своими авторами на другой. Тексты научных статей рассматриваемые в рамках данного подкорпуса были взяты из архива французских научных статей Traitement Automatique de la Langue (TALN). Подкорпус,

[51] https://joint-research-centre.ec.europa.eu/language-technology-resources/jrc-acquis_en
[52] https://www.amazon.com/

основанный на данном архиве, также содержит в себе параллельные предложения на английском и французском языках.

Статистика рассматриваемых трех подкорпусов представлена в Таблице 28.

Таблица 28 – Количество параллельных документов и предложений для представленных в [107] подкорпусов.

Подкорпус	Количество параллельных документов	Количество параллельных предложений
JRC-Acquis	≈ 10,000	≈ 150,000
APR	≈ 6,000	≈ 23,000
TALN	≈ 35	≈ 1,300

PAN-PC-2011 [111]. Корпус содержит в себе примеры как и с моноязычными заимствованиями, так и с межъязыковыми. Данные примеры были получены двумя способами: автоматической генерацией, с использованием специальной программы основанной на машинном переводе, и ручной разметкой. Для задачи поиска межъязыковых заимствований в данном корпусе содержались документы на парах языков английский-испанский и английский-французский. Те примеры, которые получались с использованием машинного перевода, были впоследствии, при необходимости, вручную исправлены. В рамках задачи сравнения представляемого алгоритма с другими из корпуса были использована только часть, содержащая в себе межъязыковые заимствования для пары языков английский-испанский.

Сгенерированный корпус. Чтобы иметь возможность сравнить работу представляемого алгоритма для разных пар языков, а также иметь возможность оценить работу алгоритма для пары английский-армянский, были сгенерированы 5 новых корпусов[53] для всех рассматриваемых языков в связке с английским. Корпус был сгенерирован по аналогии с процессом генерации корпуса CrossLang. В данном корпусе содержалось 400 анализируемых документов для каждого из 5

[53] https://drive.google.com/drive/folders/1jnAehDCQM_u1P3wKpRMozpbpiu0xbP5E?usp= sharing

языков, в которых, в отличии от CrossLang, содержалось от 0% до 80% заимствований из от 1 до 10 документов. В проверяемой коллекции содержалось 120,000 статей извлеченных из английской Wikipedia.

Детальная статистика разбиения документов по проценту межъязыковых заимствований в них представлена в Таблице 29.

Таблица 29 – Процент анализируемых документов с определенной долей заимствований в них, для всех 5 сгенерированных корпусов.

Языки	**Доля заимствований в анализируемых документах**		
	0 - 0.2	**0.2 - 0.5**	**0.5 - 0.8**
EN-HY	16.5%	65.25%	18.25%
EN-RU	15.25%	62.25%	22.5%
EN-ES	7.5%	59.0%	33.5%
EN-FR	12.0%	60.0%	28.0%
EN-DE	19.0%	63.0%	18.0%

3.3. Результаты

В рамках данного раздела производится оценка разработанного метода, а также его сравнение с другими SOTA алгоритмами обнаружения межъязыковых заимствований. Оценка и сравнение результатов производятся на представленных в *Разделе 3.2* тестовых корпусах.

3.3.1. Сравнение алгоритмов обнаружения межъязыковых заимствований

В рамках проведения экспериментов, с использованием представляемого алгоритма, каждому заимствованному предложению из анализируемого документа сопостовлялось соответствующее предложение из документа-источника. В рамках

работы алгоритма после обнаружения подобных пар предложений не предусматривается процесс пост-обработки, то есть процесс сшивки обнаруженных заимствованных предложений. Исходя из этого, сравнения различных алгоритмов в рамках данного раздела производились без использования метрики *Granularity*. Все сравнения были произведены по метрикам макро-точность и макро-полнота. Сравнения были произведены между представляемым алгоритмом и теми алгоритмами которые получили лучшие результаты для представленных в *Разделе 3.2* корпусов.

CrossLang. В рамках данного эксперимента была проверена работ алгоритма для пары языков английский-русский. Результаты представлены в Таблице 30.

Таблица 30 – Сравнение представляемого алгоритма с алгоритмом получившим лучшие результаты на англйско-русском корпусе CrossLang.

	CrossLang		
	Полнота	**Точность**	**F1**
Представляемый алгоритм	0,77	**0,86**	**0,81**
[13]	**0,79**	0,83	0,80

Параллельные корпусы. Для произведения сравнения с алгоритмом показавшим лучшие результаты на рассматриваемых трех параллельных корпусах, использовалась та же методология [37] оценки качества, что была использована лучшим алгоритмом.

В рамках данного эксперимента, так как корпусы разбиты на параллельные предложения, поиск заимствований происходит на уровне предложение к предложению. Сам процесс оценки выглядит следующим образом: для каждого предложения из одного из рассматриваемых корпусов, в качестве предложений-источников, из которых оно было заимствовано, берется параллельное данному предложению предложение и к нему случайным образом

добавляются еще 999 предложений из корпуса. Тем самым, для одного заимствованного предложения проверочной коллекцией являются 1000 предложений на другом языке из этого корпуса.

Точность считается как доля найденных предложений, из которых было произведено заимствование, на общее число возвращенных ответов алгоритма. Полнота считается как доля найденных предложений, из которых было произведено заимствований, от количества всех таких предложений. Фактически, представленные в [37] варианты подсчета точности и полноты – это те же макро-точность и макро-полнота, с одним лишь различием, что в данном случае нет смысла учитывать количество найденных символов, т.к. для каждого предложения будет либо однозначно находится все предложение, из которого было произведено заимствование, либо оно будет полностью не найдено.

Таблица 31 – Сравнение F1-мер представляемого алгоритма с алгоритмом достигающим лучших результатов на корпусах представленных в [37] для пары языков английский-французский.

	JRC-Aquis	**APR**	**TALN**
Представляемый алгоритм	$71,80 \pm 0,444$	$96,67 \pm 0,387$	$89,72 \pm 0,474$
[31]	$72,70 \pm 1,446$	$78,91 \pm 1,005$	$80,89 \pm 0,944$

Сравнение результатов, полученных с использование описанной методологии оценки алгоритмов, описаны в Таблице 31. Результаты представлены с использованием метрики F1. Дополнительно, для данных корпусов был посчитан доверительный интервал.

Исходя из результатов полученных в Таблице 31 видно, что представляемый алгоритм значительно обходит SOTA алгоритм для корпусов APR и TALN, при этом достигая сравнимых результатов для корпуса JRC-Aquis, что может быть связано с тем, что данные тексты содержат в себе юридические термины, не

предусмотренные в рамках словаря межъязыковых синонимов, представленного в *2.1.3.2*.

Таблица 32 – Сравнение представляемого алгортима с алгоритмом получившим лучшие результаты на испанско-английской части корпуса PAN-PC-11.

	PAN-PC-11 ES-EN	
	Полнота	**Точность**
Представляемый алгоритм	**0,79**	**0,85**
[39]	0,75	0,79

PAN-PC-2011. Используя часть данного корпус, была произведена оценка работы представляемого алгоритма для пары языков испанский-английский, а также произведено сравнение с алгоритмом достигающим лучших результатов, представленного в [39]. Оценка и сравнение представлены в Таблице 32. В рамках данного корпуса представляемый алгоритм также обошел SOTA.

Сгенерированный набор данных. Для оценки работы представляемого алгоритма для пары языков английский-армянский, а также для сравнения работы алгоритма относительно различных пар языков, был сгенерирован набор данных для 5 пар рассматриваемых языков. Результаты, полученные на сгенерированном корпусе, представлены в Таблице 33. В данном случае, также использовались метрики макро-полноты и макро-точности вместе с оценкой F1-меры относительно них.

Исходя из результатов полученных для сгенерированных наборов данных (Таблица 33), представляемый алгоритм показывает лучшие результаты для пар языков английский-испанский и английский-французский, что может быть связано с большим количеством слов в межъязыковом словаре синонимов для данных языков. Руководствуясь той же логикой, можно сказать, что результаты, показанные на корпусе пары языков английский-армянский, являются относительно низкими из-за малоресурсности армянского, вследствие чего в

словаре межъязыковых синонимов содержалось меньшее количество слов. В случае с парой английский-немецкий получаются худшие результаты, исходя из словообразований присущих немецкому языку с соединением простых слов и получением таким образом сложных слов. Подобного рода сложные слова не содержались в межъязыковом словаре синонимов, что негативно повлияло на общую работу представляемого алгоритма для данной пары языков.

Таблица 33 – Результаты показанные представляемым алгоритмом на сгенерированном наборе данных для 5 пар языков.

	Сгенерированный набор данных		
Языки	**Полнота**	**Точность**	**F1**
EN - HY	0,72	0,73	0,73
EN - RU	0,81	0,82	0,81
EN - ES	0,90	0,86	0,88
EN - FR	0,88	0,81	0,84
EN - DE	0,71	0,64	0,67

3.4. Выводы

Подводя итоги полученных результатов, представляемый алгоритм, основанный на использовании межъязыкового словаря синонимов, собранного с помощью модификации и дополнения мультиязычного тезауруса Universal WordNet, для извлечения кандидатов и использовании дообученной модели XLM-RoBERTa для детального анализа, достигает сравнимых, а для некоторых тестовых корпусов лучших, результатов.

Также, алгоритм показывает достойные результаты для такого малоресурсного языка, как армянский, что делает его применимым для обнаружения межъязыковых заимствований в текстах других подобных

малоресурсных языков. Применимость алгоритма к малоресурсным языкам связана с отсутствием зависимости от инструментов машинного перевода во время его работы и ненадобностью решения задачи многозначности слов с использование только самых частоиспользуемых смыслов слов. Исходя из этого, данный алгоритм может быть применен для всех языков содержащихся в тезаурусе Universal WordNet и в XLM-RoBERTa.

Однако, стоит учитывать, что данный подход имеет некоторые ограничения. В первую очередь он применим только для тех языков, для которых существуют инструменты токенизации и лемматизации. Для создания словаря межъязыковых синонимов, используемого на этапе извлечения кандидатов, применяется инструмент машинного перевода, что тоже ставит некоторые ограничения на количество языков, для которых данный алгоритм может быть применен.

Также, исходя из результатов показанных на корпусе пары языков английский-немецкий, можно сделать выводы, что алгоритм будет иметь проблемы с языками, где словообразование тесно связано с использованием сложных слов.

Дополнительно стоит также отметить, что метод был протестирован только на языках из индо-европейской языковой семьи, что делает неопределенным его совместимость с языками других языковых семей, где процессы лемматизации и токенизации могут отличаться.

Глава 4. Сравнительный анализ и слияние представляемого метода с методом представленным компанией “Антиплагиат.ру”

В рамках данной главы проводится сравнительный анализ представляемого в работе метода с методом представленным компанией “Антиплагиат.ру” в рамках [13], а также рассматриваются различные способы их слияния для нивелирования слабых мест друг друга, тем самым повысив точность нахождения межъязыковых заимствований.

В первом разделе описывается общая схема работы алгоритма обнаружения межъязыковых заимствований представленная компанией “Антиплагиат.ру”. Исходя из того, что оба метода достигали лучших результатов на различных существующих наборах данных, был сгенерирован новый тестовый набор, где в рамках одного анализируемого документа могли содержаться заимствования сразу из источников на нескольких языках. Генерация данного тестового набора данных описана во второй главе. Третий раздел посвящен экспериментам по слиянию двух методов, комбинированному и последовательному, и результатам полученным путем слияния.

4.1. Общая схема работы алгоритма обнаружения межъязыковых заимствований “Антиплагиат.ру”

Данный метод основан на использовании машинного перевода и последующем моноязычном поиске заимствований. Дополнительно, учитывается неоднозначность переводов с использованием синонимических групп и векторной модели представления текстовых фрагментов. Метод разделен на пять последовательных этапов.

4.1.1. Предобработка

Анализируемые документы и документы-источники первым делом проходят этап предобработки. Анализируемые документы подвергаются машинному переводу на язык проверочной базы с использованием инструмента Google Translate. Для переведенных анализируемых документов и документов-источников производится токенизация, удаляются стоп-слова и пунктуационные символы, а также производится стемминг.

4.1.2. Разбиение слов по синонимическим группам

На данном этапе производится замена слов на метки их синонимических групп (4.1), во избежании влияния неоднозначности перевода производимого на первом этапе.

$$\{word_1, \ldots, word_n\} \rightarrow \{class(word_1), \ldots, class(word_n)\} \quad (4.1)$$

Синонимические группы собирались с использованием различных существующих словарей синонимов. Дополнительно, данные синонимические группы были обогащены с использованием кластеризации векторных представлений слов, полученных с помощью алгортима fastText.

4.1.3. Извлечение кандидатов

Извлечение кандидатов производится с использованием метода основанного на построении инвертированного индекса поверх документов-источников. Анализируемые документы и документы-источники представляются в качестве множества шинглов на уровне слов. После чего, сравнение документов

производится с использованием данных шинглов. Для учета ситуации, при которой после произведения перевода последовательность слов изменялась, в рамках одного шингла слова сортировались в алфавитном порядке. В качестве функции оценки близости документов используется функция MinHash.

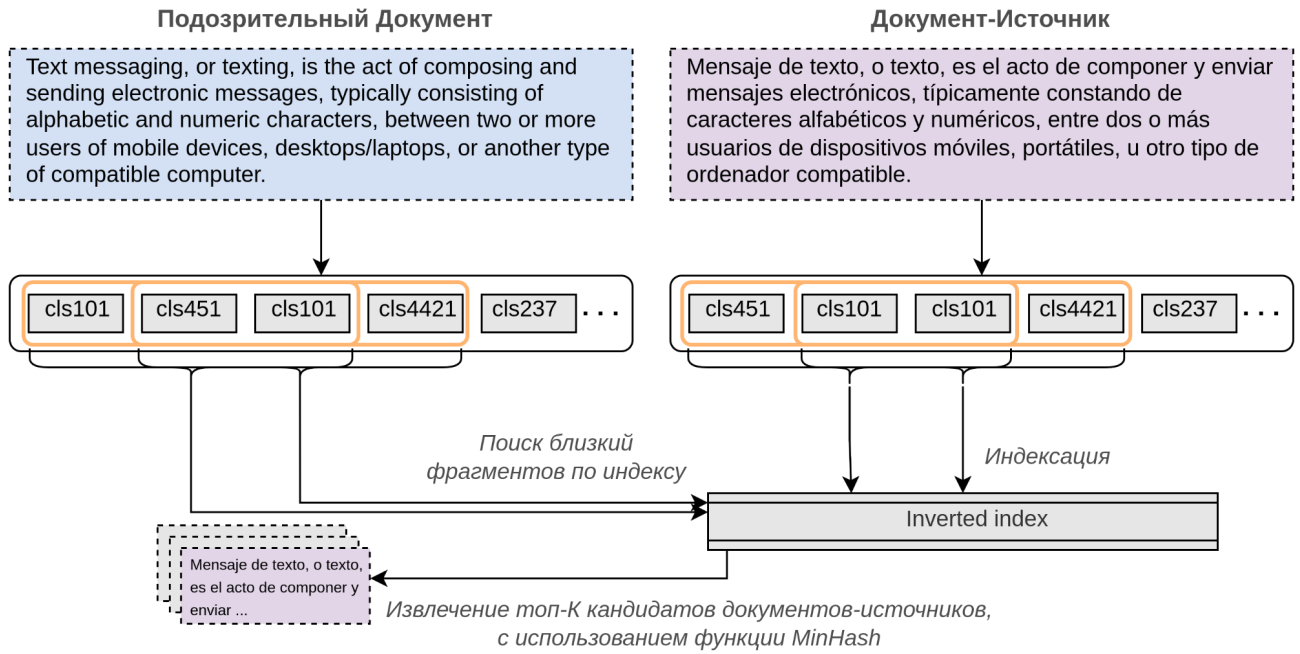

Рисунок 4.1 – Схема работы этапа извлечения кандидатов метода представленного компанией “Антиплагиат.ру”.

В конечном итоге, исходя из оценки функции MinHash, некоторое фиксированное количество документов-кандидатов с наибольшими значениями данной функции отбирались и подавались на следующий этап. Данный этап представлен на рисунке 4.1.

4.1.4. Детальный анализ

После получения топ-К кандидатов документов-источников производится этап детального анализа. В рамках данного этапа первоначально документы-кандидаты и анализируемый документ разбиваются на текстовые фрагменты некоторой длины. После чего, данные текстовые фрагменты при

помощи модели LaBSE [77] представляются в векторном виде. Далее, производится их попарное сравнение с использованием косинусной близости. Каждому фрагменту анализируемого документа сопоставляется некоторое число фрагментов-источников, прошедших определенный порог косинусной близости.

4.1.5. Генерация отчета

На данном этапе производится некоторая обработка полученных после детального анализа результатов. Некоторые фрагменты-источники, найденные в качестве источника заимствования, могут иметь между собой пересечения, быть короткими, или полностью повторяться. Во избежании таких случаев, производится этап генерации отчета, в рамках которого происходит процесс постобработки найденных фрагментов. Фрагменты соединяются, удаляются или фильтруются исходя из их длины.

4.2. Тестовый набор данных

Исходя из того, что рассматриваемые в рамках данной главы методы достигают лучших результатах на существующих различных тестовых наборах, был создан новый тестовый набор, на котором было произведено сравнение данных методов, а также протестированы различные способы слияния двух методов. Анализируемые документы данного тестового набора могут содержать в себе заимствования сразу из источников нескольких рассматриваемых языков.

Для генерации нового тестового набора была собрана коллекция из 100,000 документов-источников D. Данные документы являются случайными статьями из Wikipedia на одном из 4 языков: Армянский, Русский, Английский и Испанский. Для каждого языка содержалось 25,000 документов-источников. В качестве анализируемых документов D_{susp} использовалось по 300 документов для каждого

языка; 1200 в общем. Документы-источники и анализируемые документы не имели пересечений.

Генерация межъязыковых заимствований для каждого анализируемого документа $d_{susp} \in D_{susp}$ производилась в следующие 3 шага:

- Случайным образом выбираются от 1 до 10 документов-источников;
- Случайным образом выбираются от 20% до 60% предложений анализируемого документа;
- По количеству предложений, выбранных на втором шаге из документов-источников, выбранных на первом шаге случайным образом выбираются предложения, переводятся на нужный язык и заменяют эти выбранные предложения.

Таким образом, в рамках одного анализируемого документа могли содержатся заимствования из документов-источников на различных языках в разном количестве. В качестве инструмента перевода был использован Google Translate, как самый популярный из доступных инструментов машинного перевода.

4.3. Эксперименты по слиянию двух методов

Представляемый в данной работе метод обнаружения межъязыковых заимствований и метод представленный компанией “Антиплагиат.ру” в [13] имеют схожую структуру первых 4 этапов: предобработка (Pp), представление слов в виде меток кластеров (LIR), извлечение кандидатов (CR) и детальный анализ (DA). Используя данную схожесть структур были произведены эксперименты по слиянию двух методов.

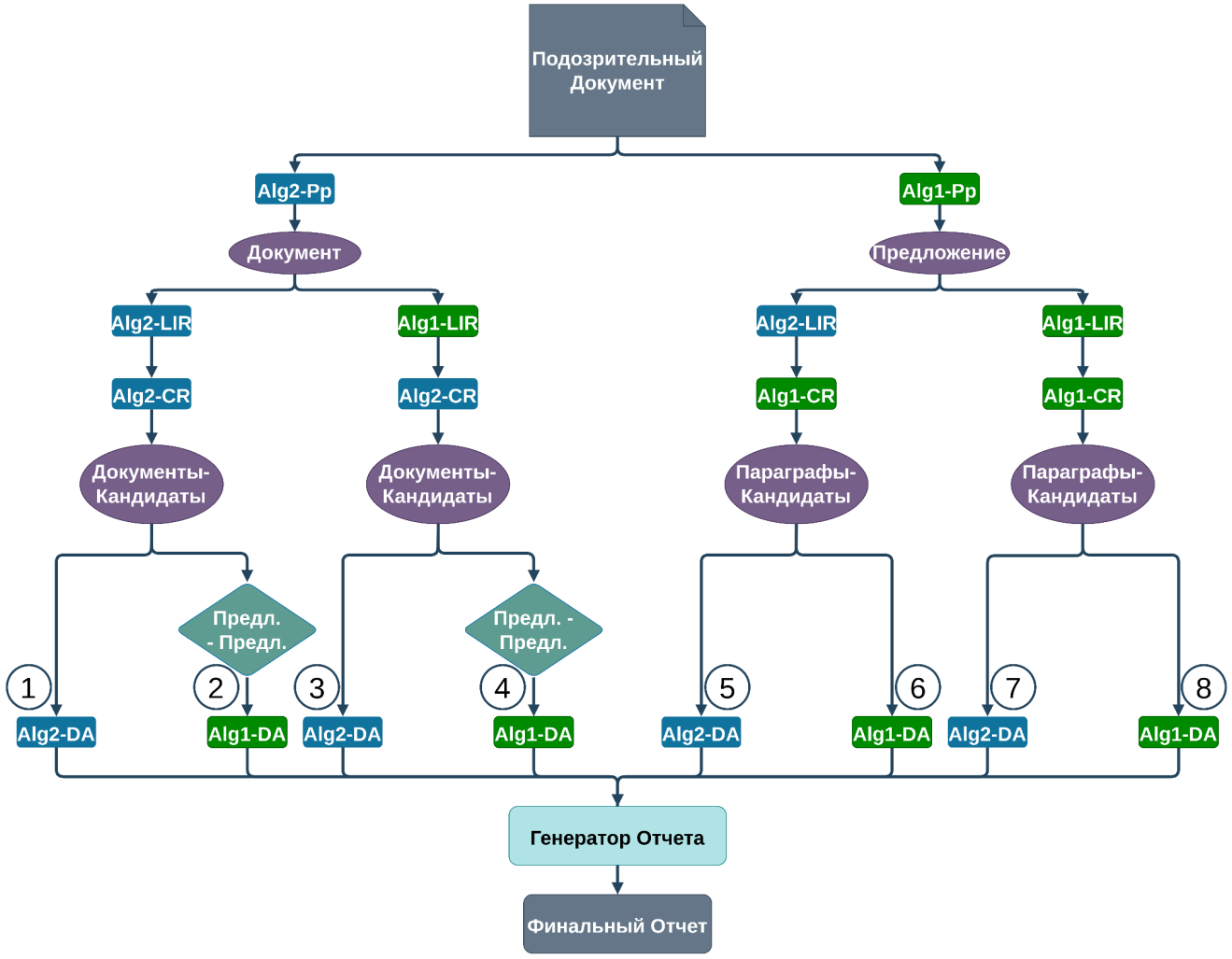

Рисунок 4.2 – Пути всех тестовых комбинаций слияния двух методов (Pp - "Предобработка", LIR - "Представление слов в виде меток кластеров", CR - "Извлечение Кандидатов", DA - "Детальный Анализ")

4.3.1. Комбинированное слияние

Исходя из схожести структур двух методов был предложен вариант замены различных этапов одного метода теми же этапами второго метода, получая таким образом различные комбинации. Таким образом, было протестировано 8 комбинаций слияния. В конце каждой из комбинаций в качестве пятого этапа использовался генератор отчета представленный в *Подразделе 4.1.5*. Все протестированные комбинации представлены на рисунке 4.2, где рассматриваемый в работе метод обозначен как “Alg1”, а метод представленный

“Антиплагиат.ру” как “Alg2”. Первая и восьмая комбинации на рисунке 4.2 отвечают за два исходных метода без смены этапов.

Блок “Предл. - Предл.” обозначает процесс спаривания предложений анализируемого документа с предложениями документов-источников. Данный процесс обусловлен тем, что этап детального анализа представляемого метода принимает на вход пары предложений, а после этапа извлечения кандидатов метода “Антиплагиат.ру” возвращаются анализируемый документ и его документы-кандидаты.

4.3.1.1. Результаты

Исходя из того, что в каждом анализируемом документе могли содержаться заимствования из трех языков,, результаты данного этапа представлены усредненными по языкам, из которых было произведено заимствование для каждого из языков анализируемых документов. Результаты полученные каждой из рассматриваемых комбинаций по аналогии с нумерацией на рисунке 4.2 представлены в Таблице 34.

Первая и восьмая комбинации, являясь первоначальными методами без замены какого-либо из их этапов, достигли лучших результатов.

Рассматриваемый в работе алгоритм достигает наивысшего значения полноты, при этом имеет плохие значения метрики Granularity. Плохие значения Granularity связаны с попарным сравнением предложений производимым на этапе детального анализа, что в некоторых случаях приводит к нахождению нескольких маленьких фрагментов в рамках одного большого фрагмента-источника. Рассматриваемый метод достигает лучших результатов по метрики F1 для всех рассматриваемых языков, а также достигает лучших значений метрики Plag Score для английского и армянского языков.

Таблица 34 – Результаты всех вариантов комбинированного слияния двух методов.

Комбинация	Язык	Точность	Полнота	Gran.	F1	Plag Score
1) Alg2-Pp→Alg2-LIR→ →Alg2-CR→Alg2-DA	Ru	**0.88**	0.30	1.0	0.45	**0.45**
	Es	**0.93**	0.42	1.0	0.58	**0.58**
	En	**0.95**	0.30	1.0	0.46	0.46
	Hy	**0.77**	0.05	1.0	0.09	0.09
2) Alg2-Pp→Alg2-LIR→ →Alg2-CR→Alg1-DA	Ru	0.59	0.10	1.0	0.17	0.17
	Es	0.62	0.10	1.0	0.18	0.18
	En	0.71	0.08	1.0	0.15	0.15
	Hy	0.19	0.02	1.0	0.04	0.04
3) Alg2-Pp→Alg1-LIR→ →Alg2-CR→Alg2-DA	Ru	0.73	0.11	1.0	0.19	0.19
	Es	0.87	0.10	1.0	0.18	0.18
	En	0.91	0.05	1.0	0.09	0.09
	Hy	0.77	0.04	1.0	0.07	0.07
4) Alg2-Pp→Alg1-LIR→ →Alg2-CR→Alg1-DA	Ru	0.40	0.06	1.0	0.11	0.11
	Es	0.38	0.06	1.07	0.11	0.10
	En	0.45	0.08	1.31	0.14	0.11
	Hy	0.18	0.02	1.0	0.04	0.04
5) Alg1-Pp→Alg2-LIR→ →Alg1-CR→Alg2-DA	Ru	0.07	0.11	1.0	0.09	0.09
	Es	0.42	0.22	1.0	0.29	0.29
	En	0.28	0.12	1.0	0.17	0.17
	Hy	0.58	0.18	1.0	0.27	0.27
6) Alg1-Pp→Alg2-LIR→ →Alg1-CR→Alg1-DA	Ru	0.44	0.77	1.53	0.56	0.42
	Es	0.44	0.74	1.61	0.55	0.40
	En	0.57	0.77	1.54	0.66	0.49
	Hy	0.33	0.51	1.27	0.40	0.34
7) Alg1-Pp→Alg1-LIR→ →Alg1-CR→Alg2-DA	Ru	0.10	0.11	1.0	0.10	0.10
	Es	0.39	0.25	1.0	0.30	0.30
	En	0.26	0.12	1.0	0.17	0.17
	Hy	0.38	0.14	1.0	0.21	0.21
8) Alg1-Pp→Alg1-LIR→ →Alg1-CR→Alg1-DA	Ru	0.45	**0.82**	1.54	**0.58**	0.43
	Es	0.46	**0.83**	1.63	**0.59**	0.42
	En	0.61	**0.87**	1.60	**0.72**	**0.52**
	Hy	0.37	**0.66**	1.41	**0.48**	**0.37**

Метод представленный “Антиплагиат.ру” достигает лучших результатов точности. Метод также имеет идеальное значение Granularity, что связано со

спецификой разбиения на фрагменты этапа детального анализа производимого в рамках данного метода. Также алгоритм достигает лучших результатов Plag Score для русского и испанского языков.

Далее произведем анализ ухудшения результатов всех вариантов комбинаций относительно исконных методов:

Комбинация 2. Alg2-Pp→Alg2-LIR→Alg2-CR→Alg1-DA. Главной слабостью данной комбинации является нужда спаривания предложений анализируемых документов и документов-источников после этапа “Alg2-CR”. Из-за большого количества пар предложений падает и общая точность, и полнота нахождения заимствованных фрагментов.

Комбинация 3. Alg2-Pp→Alg1-LIR→Alg2-CR→Alg2-DA. В рамках данной комбинации важную роль сыграло отличие в этапах предобработки, где в “Alg2-Pp” используется стемминг, который негативно влияет на этап “Alg1-LIR”, что и приводит к ухудшениям результатов.

Комбинация 4. Alg2-Pp→Alg1-LIR→Alg2-CR→Alg1-DA. Данная комбинация объединяет в себе недостатки 2 и 3 комбинаций.

Комбинация 6. Alg1-Pp→Alg2-LIR→Alg1-CR→Alg1-DA. В рамках данной комбинации происходит ухудшение за счет перевода текстов после этапа “Alg1-Pp”, что влияет на точность перевода и на последующие результаты нахождения заимствований.

Комбинация 7. Alg1-Pp→Alg1-LIR→Alg1-CR→Alg2-DA. В данном случае после этапа “Alg1-CR” на вход к “Alg2-DA” подаются пары предложение-параграф, длина которых влияет на процесс работы “Alg2-DA”, т.к. он заточен под работу с парами документов, тем самым пропускает некоторые короткие фрагменты.

Комбинация 5. Alg1-Pp→Alg2-LIR→Alg1-CR→Alg2-DA. Данная комбинация объединяет в себе недостатки 6 и 7 комбинаций.

Подводя итоги результатов полученных в рамках комбинированного слияния двух методов, можно сказать, что специфики различных этапов негативно влияют на

точность обнаружения заимствований. Исходя из этого было принято решение не заменять этапы, а просто добавить последовательно.

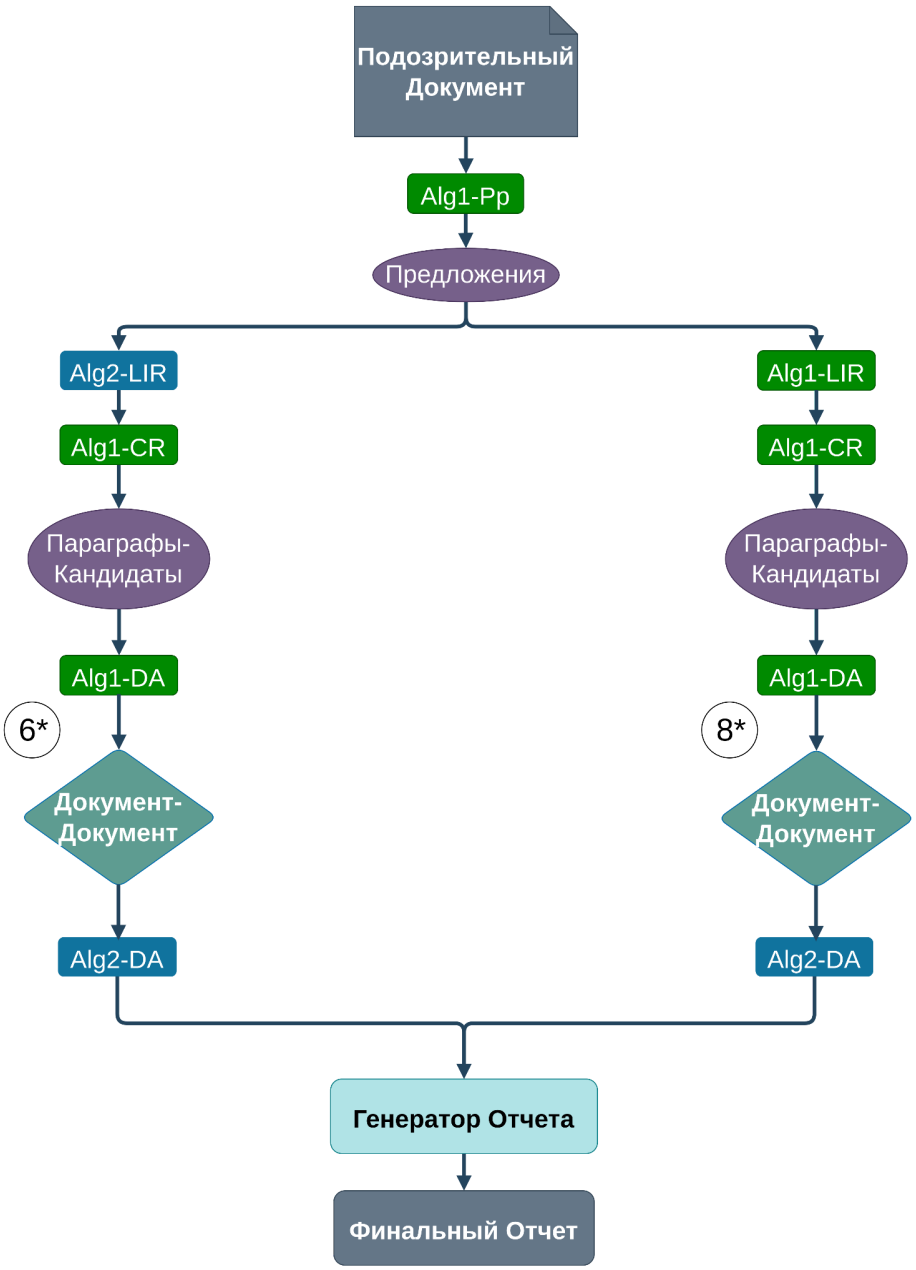

Рисунок 4.3 – Пути 6* и 8* последовательного слияния двух методов.

4.3.2. Последовательное слияние

Смотра на результаты показанные в Таблице 33, рассматриваемый в работе метод показывает высокие показатели полноты, а метод представленный

“Антиплагиат.ру” – высокие показатели точности. Пользуясь данной информацией, возникла идея дополнения рассматриваемого метода, а также “Комбинации 6”, где достигаются наивысшие показатели полноты, этапом детального анализа из метода “Антиплагиат.ру” (рисунок 4.3). Данное дополнение привело бы к повышению точности с минимальными потерями для полноты.

После этапа “Alg1-DA” возвращаются пары анализируемое предложение-предложение-источник, которые обеспечивают высокое значение полноты. Учитывая высокую точность, показываемую при использовании “Alg2-DA”, и его заточенность под работу с парами документов, перед подачей результатов “Alg1-DA” в “Alg2-DA”, производился дополнительный этап. Каждому анализируемому документу сопоставляются документы всех предложений-источников полученных после этапа “Alg1-DA”. Тем самым на вход в “Alg2-DA” подаются пары документов.

Полученные в рамках последовательного слияния, усредненные по языкам источников, результаты представлены в Таблице 35.

Таблица 35 – Результаты последовательного слияния двух методов.

Слияние	**Язык**	**Точн.**	**Полн.**	**Gran.**	**F1**	**Plag Score**
6*) Alg1-Pp→Alg2-LIR→Alg1-CR →Alg1-DA→Alg2-DA	Ru	0.97	0.62	1.0	0.76	0.76
	Es	0.98	0.64	1.0	0.77	0.77
	En	0.99	0.63	1.0	0.77	0.77
	Hy	0.90	0.34	1.0	0.49	0.49
8*) Alg1-Pp→Alg1-LIR→Alg1-CR →Alg1-DA→Alg2-DA	Ru	0.97	0.66	1.0	0.78	**0.78**
	Es	0.98	0.71	1.0	0.83	**0.83**
	En	0.99	0.69	1.0	0.81	**0.81**
	Hy	0.90	0.45	1.0	0.60	**0.60**

4.3.2.1. Дополнительная статистика по результатам последовательного слияния

В рамках данного подраздела описывается некоторая дополнительная статистика для последовательного слияния 8* получившего лучшие результаты.

Таблица 36 – Результаты достигаемые последовательным слиянием 8* для каждой пары языков.

Язык Анл.	Язык Источ.	MLFF	MLNFF	Точн.	Полн.	Gran.	F1	Plag Score
Ru	Es	254.3	206.4	0.97	0.78	1.0	0.87	0.87
	En	273.9	249.5	0.98	0.77	1.0	0.87	0.87
	Hy	238.4	230.9	0.94	0.42	1.0	0.58	0.58
Es	Ru	297.3	297.6	0.98	0.80	1.0	0.88	0.88
	En	294.0	265.8	0.99	0.87	1.0	0.93	0.93
	Hy	249.7	253.0	0.97	0.49	1.002	0.65	0.65
En	Ru	281.8	264.6	0.99	0.81	1.0	0.89	0.89
	Es	255.2	214.6	0.99	0.84	1.0	0.91	0.91
	Hy	249.9	242.9	0.98	0.45	1.002	0.62	0.61
Hy	Ru	312.8	267.0	0.87	0.38	1.0	0.53	0.53
	Es	260.1	174.5	0.88	0.53	1.0	0.66	0.66
	En	286.2	253.6	0.94	0.44	1.0	0.60	0.60

В Таблице 36 представлены результаты 8* по отдельности для каждой пары языков (т.е. Hy→Ru обозначает результаты нахождения фрагментов русского языка в текстах армянского языка). Исходя из полученных результатов можно сделать вывод, что при работе с документами на армянском языке, точность работы метода ниже чем для других языков и достигает примерно 60% метрик F1 и Plag Score. Ухудшение результатов относительно других языков связано с недостаточно адаптированными под армянский язык моделей XLM-RoBERTa и LaBSE, а также с более меньшим и менее качественным представление слов в

словаре “межъязыковых синонимов”. Для остальных же языков метрики F1 и Plag Score равны значениям в районе 90%.

В дополнение к результатам была также посчитана статистика показывающая зависимость нахождения заимствования от размера заимствованного фрагмента. В Таблице 36 представлены среднее количество символов на каждый найденный заимствованный фрагмент (MLFF) и среднее количество символов на каждый не найденный заимствованный фрагмент (MLNFF). Таким образом, в среднем, более длинные фрагменты находятся чаще более коротких, что происходит из-за большей информации которую в себе содержит длинный фрагмент текста.

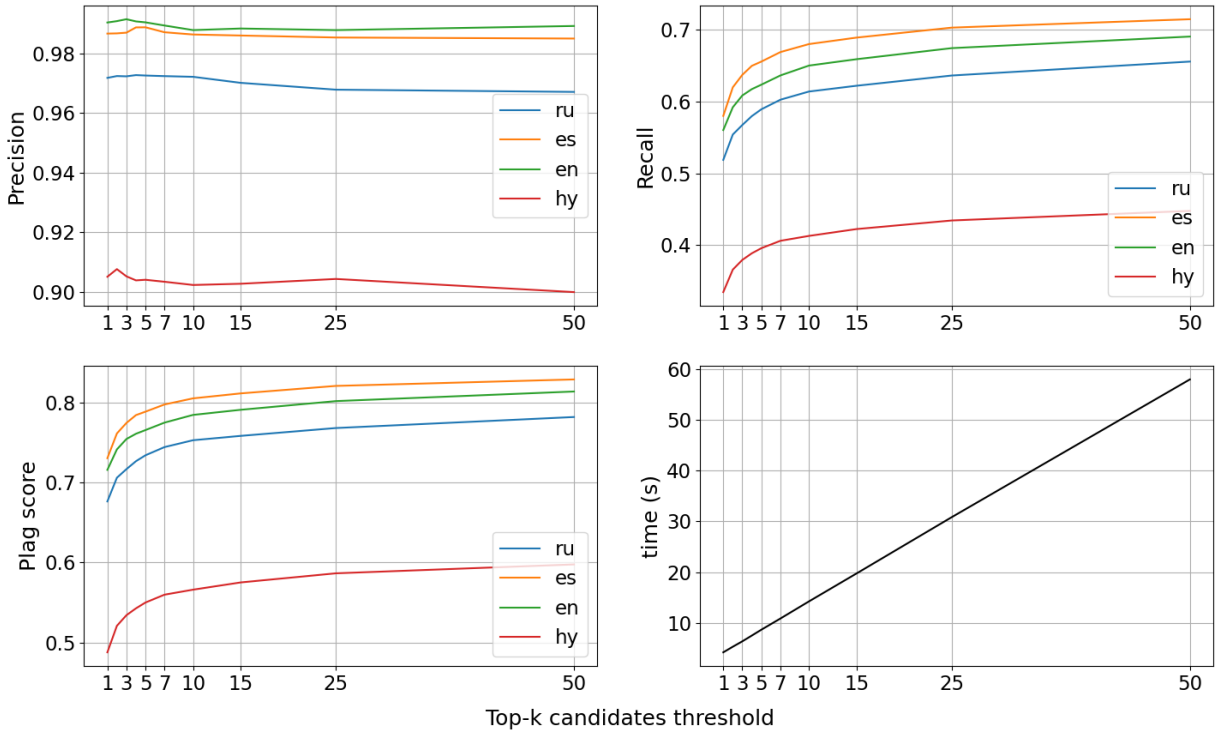

Рисунок 4.4 – Зависимость метрик качества и вычислительной сложности от изменения числа k возвращаемых параграфов-кандидатов на этапе извлечения кандидатов.

Так как на этапе извлечения кандидатов представляемого в работе метода используется гиперпараметр k отвечающий за количество возвращаемых параграфов-кандидатов для каждого анализируемого предложения, дополнительно

была посчитана статистика зависимости метрик обнаружения заимствований, а также временной сложности от значения k. Гиперпараметр k изменялся в пределах от 1 до 50 (все результаты описанные выше были получены при значении $k = 50$). Результаты полученные при изменении k представлены на рисунке 4.4. Временная сложность показана для обработки одного анализируемого документа. Можно утверждать, что примерно к значению $k = 50$ метрики выходят на плато, и дальнейшее увеличение значения k не даст значительного прироста. Вычислительная сложность растет линейно относительно значения k. Дополнительно в Таблице 37 произведено сравнение рассматриваемого в работе метода, метода “Антиплагиат.ру” и метода последовательного слияния с точки зрения временной сложности для обработки одного анализируемого документа.

Таблица 37 – Временная сложность рассматриваемого в работе метода, метода “Антиплагиат.ру” и метода последовательного слияния 8*, для обработки одного анализируемого документа.

Alg1	**Alg2**	**Слияние 8***
56,5 sec.	19,2 sec.	57,9 sec.

4.4. Выводы

В данной главе был произведен сравнительный анализ представляемого в работе метода с методом представленным компанией “Антиплагиат.ру”, а также рассмотрены различные способы их слияния для нивелирования слабых мест друг друга, тем самым повышая точность нахождения межъязыковых заимствований. Также был представлен новый тестовый набор данных, в котором каждый анализируемый документ может содержать заимствования сразу из нескольких языков. Дополнительно представляемый метод был улучшен за счет последовательного слияния с ним этапа детального анализа представленного в методе “Антиплагиат.ру”. На примере армянского языка было показано, что

улучшенный метод обнаружения межъязыковых заимствований также применим к малоресурсным языкам. Результаты описанные в рамках данной главы представлены в [3].

Заключение

Основные результаты работы заключаются в следующем:

1. Разработан новый метод обнаружения межъязыковых заимствований, превосходящий по эффективности существующие. Дополнительно, метод применим к задаче обнаружения межъязыковых заимствований в текстах малоресурсных языков.
2. Разработан новый метод генерации словаря “межъязыковых синонимов”, позволяющего достичь высоких показателей метрики полноты для этапа извлечения кандидатов в задаче обнаружения межъязыковых текстовых заимствований.
3. Разработан новый метод генерации искусственных атак “черного ящика” на языковые модели бинарной классификации, превосходящий по доле успешных атак, а также по дистанции Левенштейна и семантической близости все существующие аналоги.
4. Разработана методика выбора языковой модели для этапа детального анализа учитывающая угрозу возможности осуществления искусственных атак.

Список литературы

1. Avetisyan K., Malajyan A., Ghukasyan T. A Simple and Effective Method of Cross-Lingual Plagiarism Detection //arXiv preprint arXiv:2304.01352. – 2023.
2. Аветисян К.И., Асатрян А.А., Гукасян Ц.Г., Ешилбашян Е.М., Маладжян А.А., Недумов Я.Р., Скорняков К.А., Тигранян Ш.Т., Турдаков Д.Ю. «Sieve» / Свидетельство о государственной регистрации программы для ЭВМ, рег. №2021668213 от 11.11.2021 - Российская Федерация, 2021.
3. Avetisyan K., Gritsay G., Grabovoy A. Cross-Lingual Plagiarism Detection: Two Are Better Than One //Programming and Computer Software. – 2023. – Т. 49. – №. 4. – С. 346-354.
4. Ghukasyan T., Yeshilbashyan Y., Avetisyan K. Subwords-only alternatives to fastText for morphologically rich languages //Programming and Computer Software. – 2021. – Т. 47. – С. 56-66.
5. Ter-Hovhannisyan T., Avetisyan K. Transformer-Based Multilingual Language Models in Cross-Lingual Plagiarism Detection //2022 Ivannikov Memorial Workshop (IVMEM). – IEEE, 2022. – С. 72-80.
6. Avetisyan K. Dialects Identification of Armenian Language //Proceedings of the Workshop on Processing Language Variation: Digital Armenian (DigitAm) within the 13th Language Resources and Evaluation Conference. – 2022. – С. 8-12.
7. Potthast M., Stein B., Eiselt A., Rosso A. B.-C. P. Overview of the 1st international competition on plagiarism detection // 3rd PAN Workshop. Uncovering Plagiarism. — Authorship and Social Software Misuse. — 2009. — С. 1-9.
8. Potthast M., Barrón-Cedeño A., Eiselt A., Stein B., Rosso P. Overview of the 2nd International Competition on Plagiarism Detection // Working Notes Papers of the CLEF 2010 Evaluation Labs. — Lecture Notes in Computer Science. — 2010. — Vol. 1176.

9. Potthast M. , Eiselt A., Barrón-Cedeño L. A., Stein B., Rosso P. Overview of the 3rd international competition on plagiarism detection // CEUR workshop proceedings. – CEUR Workshop Proceedings — 2011. – T. 1177.
10. Kent C. K., Salim N. Web based cross language plagiarism detection // 2010 Second International Conference on Computational Intelligence, Modelling and Simulation. – IEEE, 2010. – C. 199-204. — DOI: https://doi.org/10.48550/arXiv.0912.3959.
11. Sanchez-Perez M. A. , Sidorov G., Gelbukh A. F., A Winning Approach to Text Alignment for Text Reuse Detection at PAN 2014 // CLEF (Working Notes). – 2014. – T. 2014. – C. 1004-1011.
12. Muneer I. et al. CLEU‐A Cross‐language english‐urdu corpus and benchmark for text reuse experiments // Journal of the Association for Information Science and Technology. – 2019. – T. 70. – №. 7. – C. 729-741.
13. Bakhteev O. et al. CrossLang: the system of cross-lingual plagiarism detection // Workshop on Document Intelligence at NeurIPS. – 2019.
14. Kuznetsova M. V., Bakhteev O. Y., Chekhovich Y. V. Methods of cross-lingual text reuse detection in large textual collections // Informatika I Ee Primeneniya [Informatics and Its Applications]. – 2021. – T. 15. – №. 1. – C. 30-41. — DOI: https://doi.org/10.14357/19922264210105.
15. Martin B. Teach You Backwards: An In-Depth Study of Google Translate for 108 Languages – 2019. – URL: https://www.teachyoubackwards.com/empirical-evaluation/.
16. Potthast M., Stein B., Anderka M. A wikipedia-based multilingual retrieval model // Advances in Information Retrieval: 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings 30. – Springer Berlin Heidelberg, 2008. – C. 522-530.
17. Barrón-Cedeno A. et al. On Cross-lingual Plagiarism Analysis using a Statistical Model // PAN. – 2008. – T. 212. – C. 1-10.

18. Franco-Salvador M., Rosso P., Montes-y-Gómez M. A systematic study of knowledge graph analysis for cross-language plagiarism detection // Information Processing & Management. – 2016. – T. 52. – №. 4. – C. 550-570.
19. Pasini T., Raganato A., Navigli R. XL-WSD: An extra-large and cross-lingual evaluation framework for word sense disambiguation //Proceedings of the AAAI Conference on Artificial Intelligence. – 2021. – T. 35. – №. 15. – C. 13648-13656.
20. Scarlini B., Pasini T., Navigli R. Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation //Proceedings of the AAAI conference on artificial intelligence. – 2020. – T. 34. – №. 05. – C. 8758-8765.
21. Procopio L. et al. MultiMirror: Neural Cross-lingual Word Alignment for Multilingual Word Sense Disambiguation //IJCAI. – 2021. – C. 3915-3921.
22. Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.
23. Vaswani A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – T. 30.
24. Potthast M. et al. Cross-language plagiarism detection // Language Resources and Evaluation. – 2011. – T. 45. – C. 45-62. – DOI: https://doi.org/10.1007/s10579-009-9114-z.
25. McNamee P., Mayfield J. Character n-gram tokenization for European language text retrieval // Information retrieval. – 2004. – T. 7. – C. 73-97.
26. Gabrilovich E. et al. Computing semantic relatedness using Wikipedia-based explicit semantic analysis // IJcAI. – 2007. – T. 7. – C. 1606-1611.
27. Brown P. F. et al. The mathematics of statistical machine translation: Parameter estimation. – 1993.
28. Brown P. F. et al. A statistical approach to machine translation //Computational linguistics. – 1990. – T. 16. – №. 2. – C. 79-85.
29. Civera J., Juan A. Mixtures of ibm model 2 //Proceedings of the 11th Annual conference of the European Association for Machine Translation. – 2006.

30. Navigli R., Ponzetto S. P. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network //Artificial intelligence. – 2012. – Т. 193. – С. 217-250.
31. Vossen P. Introduction to eurowordnet // EuroWordNet: A multilingual database with lexical semantic networks. – 1998. – С. 1-17.
32. Ceska Z., Toman M., Jezek K. Multilingual plagiarism detection // Artificial Intelligence: Methodology, Systems, and Applications: 13th International Conference, AIMSA 2008, Varna, Bulgaria, September 4-6, 2008. Proceedings 13. – Springer Berlin Heidelberg, 2008. – С. 83-92.
33. Gupta P., Barrón-Cedeno A., Rosso P. Cross-language high similarity search using a conceptual thesaurus //Information Access Evaluation. Multilinguality, Multimodality, and Visual Analytics: Third International Conference of the CLEF Initiative, CLEF 2012, Rome, Italy, September 17-20, 2012. Proceedings 3. – Springer Berlin Heidelberg, 2012. – С. 67-75. – DOI: https://doi.org/10.1007/978-3-642-33247-0_8.
34. Franco-Salvador M., Gupta P., Rosso P. Cross-language plagiarism detection using a multilingual semantic network //Advances in Information Retrieval: 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings 35. – Springer Berlin Heidelberg, 2013. – С. 710-713. – DOI: https://doi.org/10.1007/978-3-642-36973-5_66.
35. Franco-Salvador M., Rosso P., Montes-y-Gómez M. A systematic study of knowledge graph analysis for cross-language plagiarism detection //Information Processing & Management. – 2016. – Т. 52. – №. 4. – С. 550-570. – DOI: https://doi.org/10.1016/j.ipm.2015.12.004.
36. Roostaee M., Sadreddini M. H., Fakhrahmad S. M. An effective approach to candidate retrieval for cross-language plagiarism detection: A fusion of conceptual and keyword-based schemes //Information Processing & Management. – 2020. – Т. 57. – №. 2. – С. 102150. – DOI: https://doi.org/10.1016/j.ipm.2019.102150.

37. Ferrero J. et al. Using word embedding for cross-language plagiarism detection // arXiv preprint arXiv:1702.03082. – 2017.
38. Bérard A. et al. MultiVec: a multilingual and multilevel representation learning toolkit for NLP // The 10th edition of the Language Resources and Evaluation Conference (LREC). – 2016.
39. Roostaee M., Fakhrahmad S. M., Sadreddini M. H. Cross-language text alignment: A proposed two-level matching scheme for plagiarism detection // Expert Systems with Applications. – 2020. – T. 160. – C. 113718.
40. Lample G. et al. Unsupervised machine translation using monolingual corpora only // arXiv preprint arXiv:1711.00043. – 2017.
41. Conneau A. et al. Word translation without parallel data // arXiv preprint arXiv:1710.04087. – 2017.
42. Zubarev D., Sochenkov I. Cross-language text alignment for plagiarism detection based on contextual and context-free models // Proceedings of the Annual International Conference “Dialogue. – 2019. – T. 1. – C. 799-810.
43. Conneau A. et al. Unsupervised cross-lingual representation learning at scale // arXiv preprint arXiv:1911.02116. – 2019.
44. Robertson S. E. et al. Okapi at TREC-3 //Nist Special Publication Sp. – 1995. – T. 109. – C. 109.
45. De Melo G., Weikum G. Towards a universal wordnet by learning from combined evidence //Proceedings of the 18th ACM conference on Information and knowledge management. – 2009. – C. 513-522.
46. De Melo G., Weikum G. MENTA: Inducing multilingual taxonomies from Wikipedia //Proceedings of the 19th ACM international conference on Information and knowledge management. – 2010. – C. 1099-1108.
47. De Melo G., Weikum G. Constructing and utilizing wordnets using statistical methods //Language Resources and Evaluation. – 2012. – T. 46. – C. 287-311.
48. Shavrina T., Shapovalova O. To the methodology of corpus construction for machine learning:“Taiga” syntax tree corpus and parser //Proceedings of “CORPORA-2017” International Conference. – 2017. – C. 78-84.

49. Mikolov T. et al. Advances in pre-training distributed word representations //arXiv preprint arXiv:1712.09405. – 2017.
50. Grave E. et al. Learning word vectors for 157 languages //arXiv preprint arXiv:1802.06893. – 2018.
51. Pennington J., Socher R., Manning C. D. Glove: Global vectors for word representation //Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). – 2014. – С. 1532-1543.
52. Avetisyan K., Ghukasyan T. Word embeddings for the armenian language: intrinsic and extrinsic evaluation //arXiv preprint arXiv:1906.03134. – 2019.
53. Straka M., Hajic J., Straková J. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, pos tagging and parsing //Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). – 2016. – С. 4290-4297.
54. Straka M. UDPipe 2.0 prototype at CoNLL 2018 UD shared task //Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. – 2018. – С. 197-207.
55. Sérasset G. DBnary: Wiktionary as a Lemon-based multilingual lexical resource in RDF //Semantic Web. – 2015. – Т. 6. – №. 4. – С. 355-361.
56. Miller G. A. WordNet: a lexical database for English //Communications of the ACM. – 1995. – Т. 38. – №. 11. – С. 39-41.
57. Miller G. A. WordNet: An electronic lexical database. – MIT press, 1998.
58. Princeton University "About WordNet." – WordNet. –- Princeton University. – 2010. – URL: https://wordnet.princeton.edu/.
59. Qi P. et al. Stanza: A Python natural language processing toolkit for many human languages //arXiv preprint arXiv:2003.07082. – 2020.
60. Manning C. D. et al. The Stanford CoreNLP natural language processing toolkit //Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations. – 2014. – С. 55-60.
61. Kim Y. et al. Character-aware neural language models //Proceedings of the AAAI conference on artificial intelligence. – 2016. – Т. 30. – №. 1.

62. Hochreiter S., Schmidhuber J. Long short-term memory //Neural computation. – 1997. – Т. 9. – №. 8. – С. 1735-1780.
63. Pataki M. A new approach for searching translated plagiarism. – 2012.
64. Truica C. O., Velcin J., Boicea A. Automatic language identification for romance languages using stop words and diacritics //2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). – IEEE, 2015. – С. 243-246.
65. Xue L. et al. mT5: A massively multilingual pre-trained text-to-text transformer //arXiv preprint arXiv:2010.11934. – 2020.
66. Malajyan A., Avetisyan K., Ghukasyan T. Arpa: Armenian paraphrase detection corpus and models //2020 Ivannikov Memorial Workshop (IVMEM). – IEEE, 2020. – С. 35-39.
67. Peinelt N., Nguyen D., Liakata M. tBERT: Topic models and BERT joining forces for semantic similarity detection //Proceedings of the 58th annual meeting of the association for computational linguistics. – 2020. – С. 7047-7055.
68. Gangadharan V. et al. Paraphrase detection using deep neural network based word embedding techniques //2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184). – IEEE, 2020. – С. 517-521.
69. Pires T., Schlinger E., Garrette D. How multilingual is multilingual BERT? //arXiv preprint arXiv:1906.01502. – 2019.
70. Sanh V. et al. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter //arXiv preprint arXiv:1910.01108. – 2019.
71. Reimers N., Gurevych I. Sentence-bert: Sentence embeddings using siamese bert-networks //arXiv preprint arXiv:1908.10084. – 2019.
72. Reimers N., Gurevych I. Making monolingual sentence embeddings multilingual using knowledge distillation //arXiv preprint arXiv:2004.09813. – 2020.
73. Schwenk H. et al. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia //arXiv preprint arXiv:1907.05791. – 2019.
74. Dolan B., Brockett C. Automatically constructing a corpus of sentential paraphrases //Third International Workshop on Paraphrasing (IWP2005). – 2005.

75. Antonova A., Misyurev A. Building a web-based parallel corpus and filtering out machine-translated text //Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web. – 2011. – С. 136-144.
76. Cer D. et al. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation //arXiv preprint arXiv:1708.00055. – 2017.
77. Yang Y. et al. Multilingual universal sentence encoder for semantic retrieval //arXiv preprint arXiv:1907.04307. – 2019.
78. Feng F. et al. Language-agnostic bert sentence embedding //arXiv preprint arXiv:2007.01852. – 2020.
79. Heffernan K., Çelebi O., Schwenk H. Bitext mining using distilled sentence representations for low-resource languages //arXiv preprint arXiv:2205.12654. – 2022.
80. Lin T. Y. et al. Focal loss for dense object detection //Proceedings of the IEEE international conference on computer vision. – 2017. – С. 2980-2988.
81. Smith L. N. Cyclical learning rates for training neural networks //2017 IEEE winter conference on applications of computer vision (WACV). – IEEE, 2017. – С. 464-472.
82. Goodfellow I. J., Shlens J., Szegedy C. Explaining and harnessing adversarial examples //arXiv preprint arXiv:1412.6572. – 2014.
83. Kurakin A., Goodfellow I. J., Bengio S. Adversarial examples in the physical world //Artificial intelligence safety and security. – Chapman and Hall/CRC, 2018. – С. 99-112.
84. Chakraborty A. et al. Adversarial attacks and defences: A survey //arXiv preprint arXiv:1810.00069. – 2018.
85. Zhang X., Zhao J., LeCun Y. Character-level convolutional networks for text classification //Advances in neural information processing systems. – 2015. – Т. 28.
86. Pang B., Lee L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales //arXiv preprint cs/0506075. – 2005. – DOI: https://doi.org/10.3115/1219840.1219855.

87. Ebrahimi J. et al. Hotflip: White-box adversarial examples for text classification //arXiv preprint arXiv:1712.06751. – 2017.
88. Ebrahimi J., Lowd D., Dou D. On adversarial examples for character-level neural machine translation //arXiv preprint arXiv:1806.09030. – 2018.
89. Sun L. et al. Adv-bert: Bert is not robust on misspellings! generating nature adversarial samples on bert //arXiv preprint arXiv:2003.04985. – 2020.
90. Garg S., Ramakrishnan G. Bae: Bert-based adversarial examples for text classification //arXiv preprint arXiv:2004.01970. – 2020.
91. Zhao X. et al. Generating Textual Adversaries with Minimal Perturbation //arXiv preprint arXiv:2211.06571. – 2022.
92. Alzantot M. et al. Generating natural language adversarial examples //arXiv preprint arXiv:1804.07998. – 2018.
93. Ren S. et al. Generating natural language adversarial examples through probability weighted word saliency //Proceedings of the 57th annual meeting of the association for computational linguistics. – 2019. – C. 1085-1097. –DOI: https://doi.org/10.18653/v1/P19-1103.
94. Zang Y. et al. Word-level textual adversarial attacking as combinatorial optimization //arXiv preprint arXiv:1910.12196. – 2019. – DOI: https://doi.org/10.18653/v1/2020.acl-main.540.
95. Jia R. et al. Certified robustness to adversarial word substitutions //arXiv preprint arXiv:1909.00986. – 2019.
96. Li L. et al. Bert-attack: Adversarial attack against bert using bert //arXiv preprint arXiv:2004.09984. – 2020.
97. Belinkov Y., Bisk Y. Synthetic and natural noise both break neural machine translation //arXiv preprint arXiv:1711.02173. – 2017.
98. Gao J. et al. Black-box generation of adversarial text sequences to evade deep learning classifiers //2018 IEEE Security and Privacy Workshops (SPW). – IEEE, 2018. – C. 50-56.

99. Grainger J., Whitney C. Does the huamn mnid raed wrods as a wlohe? //Trends in cognitive sciences. – 2004. – Т. 8. – №. 2. – С. 58-59. – DOI: https://doi.org/10.1016/j.tics.2003.11.006.
100. Jin D. et al. Is bert really robust? a strong baseline for natural language attack on text classification and entailment //Proceedings of the AAAI conference on artificial intelligence. – 2020. – Т. 34. – №. 05. – С. 8018-8025. – DOI: https://doi.org/10.1609/aaai.v34i05.6311.
101. Song X. et al. Fast wordpiece tokenization //arXiv preprint arXiv:2012.15524. – 2020.
102. Dong Z., Dong Q. HowNet-a hybrid language and knowledge resource //International conference on natural language processing and knowledge engineering, 2003. Proceedings. 2003. – IEEE, 2003. – С. 820-824.
103. Honnibal M., Montani I. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing //To appear. – 2017. – Т. 7. – №. 1. – С. 411-420.
104. Korobov M. Morphological analyzer and generator for Russian and Ukrainian languages //Analysis of Images, Social Networks and Texts: 4th International Conference, AIST 2015, Yekaterinburg, Russia, April 9–11, 2015, Revised Selected Papers 4. – Springer International Publishing, 2015. – С. 320-332.
105. Cer D. et al. Universal sentence encoder //arXiv preprint arXiv:1803.11175. – 2018.
106. Li J. et al. Textbugger: Generating adversarial text against real-world applications //arXiv preprint arXiv:1812.05271. – 2018.
107. Potthast M. et al. An evaluation framework for plagiarism detection //Coling 2010: Posters. – 2010. – С. 997-1005.
108. Ferrero J. et al. A multilingual, multi-style and multi-granularity dataset for cross-language textual similarity detection //Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). – 2016. – С. 4162-4169.

109. Prettenhofer P., Stein B. Cross-language text classification using structural correspondence learning //Proceedings of the 48th annual meeting of the association for computational linguistics. – 2010. – С. 1118-1127.

110. Boudin F. TALN Archives: a digital archive of French research articles in Natural Language Processing (TALN Archives: une archive numérique francophone des articles de recherche en Traitement Automatique de la Langue)[in French] //Proceedings of TALN 2013 (Volume 2: Short Papers). – 2013. – С. 507-514.

111. Potthast M. , Stein B.,. Eiselt A, Barrón-Cedeño A., Rosso P. Pan plagiarism corpus 2011 (pan-pc-11). – 2011. – DOI: https://doi.org/10.5281/zenodo.3250095.

Приложение А

Результаты различных языковых моделей на сложных тестовых выборках при использовании 1% и 10% обучающих данных

Рисунок А.1 – Значения оценки F1-macro на каждой тестовой выборке достигнутые дообученными на 10% данных 14 обучающих выборок моделями. Ось X каждого из графиков обозначает обучающую выборку на которой была дообучена модель. Графики разбиты по языковым группам тестовых наборов. Для каждой тестовой выборки результаты показаны только для моделей содержащих в процессе дообучения язык тестовой выборки.

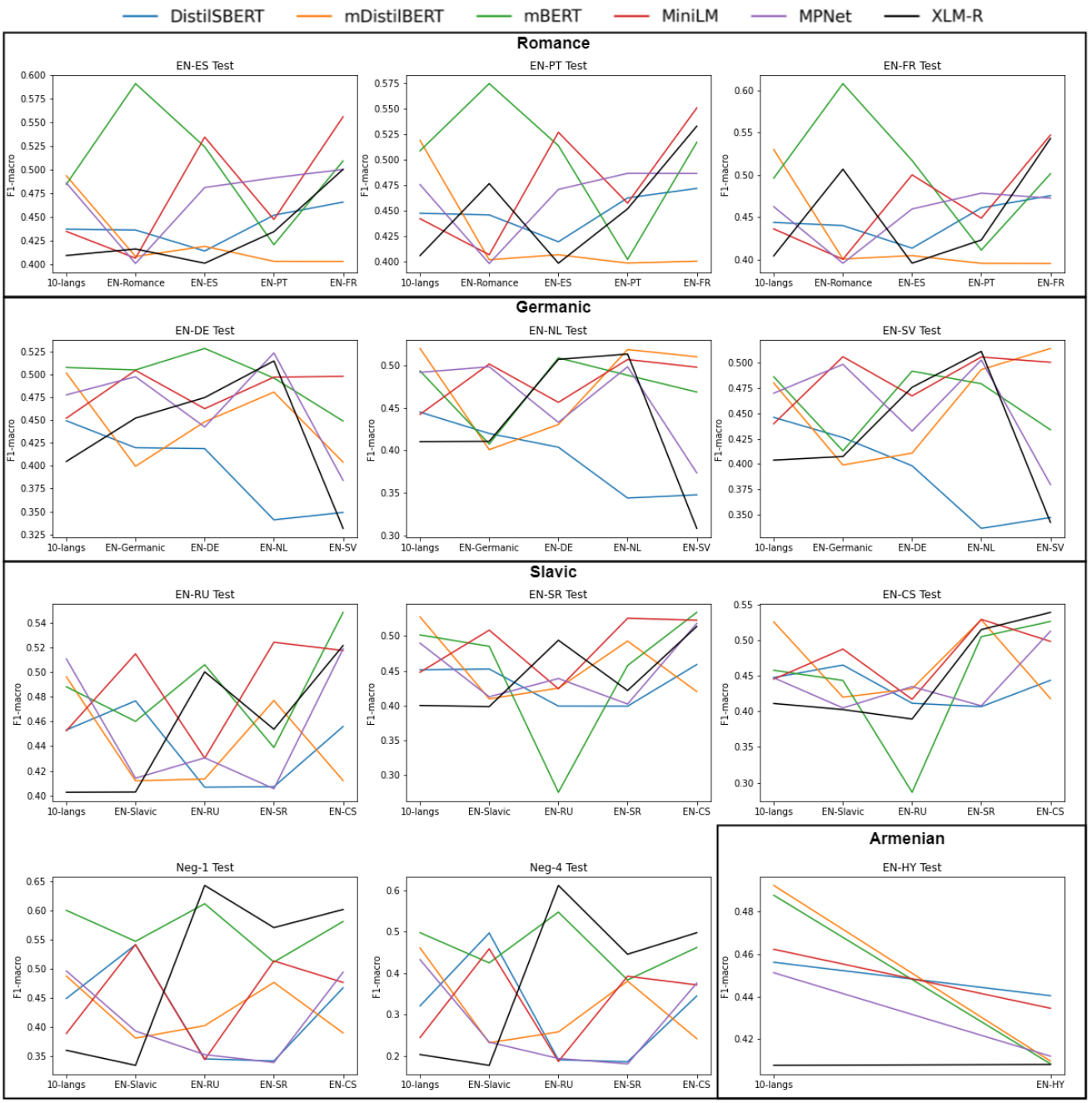

Рисунок А.2 – Значения оценки F1-macro на каждой тестовой выборке достигнутые дообученными на 10% данных 14 обучающих выборок моделями. Ось X каждого из графиков обозначает обучающую выборку на которой была дообучена модель. Графики разбиты по языковым группам тестовых наборов. Для каждой тестовой выборки результаты показаны только для моделей содержащих в процессе дообучения язык тестовой выборки.

Приложение Б

Языковая переносимость рассматриваемых 6 моделей для славянской и германской групп языков

Таблица 38 – Оценка F1-macro для каждой модели обученной на одном из языков славянской группы и протестированной на другом языке той же группы с использованием 100% данных во время дообучения.

Модели	**Обучающая выборка**	**Тестовая выборка**		
		EN - RU	EN - SR	EN - CS
mBERT	EN - RU	-	0,5129	0,4254
	EN - SR	0,6278	-	0,5281
	EN - CS	0,607	0,6016	-
mDistilBERT	EN - RU	-	0,5383	0,4084
	EN - SR	0,5437	-	0,4752
	EN - CS	0,4537	0,4262	-
XLM-RoBERTa	EN - RU	-	0,6623	0,6319
	EN - SR	0,6691	-	0,6494
	EN - CS	0,6641	0,6536	-
MiniLM	EN - RU	-	0,5806	0,5615
	EN - SR	0,624	-	0,5563
	EN - CS	0,579	0,5358	-
MPNET	EN - RU	-	0,5434	0,5578
	EN - SR	0,6005	-	0,568
	EN - CS	0,5372	0,4718	-
DistilSBERT	EN - RU	-	0,5735	0,574
	EN - SR	0,5985	-	0,5751
	EN - CS	0,5684	0,5551	-

Таблица 39 – Оценка F1-macro для каждой модели обученной на одном из языков германской группы и протестированной на другом языке той же группы с использованием 100% данных во время дообучения.

Модели	**Обучающая выборка**	**Тестовая выборка**		
		EN - DE	EN - NL	EN - SV
mBERT	EN - DE	-	0,5848	0,5973
	EN - NL	0,5665	-	0,5255
	EN - SV	0,592	0,5878	-
mDistilBERT	EN - DE	-	0,5241	0,5237
	EN - NL	0,559	-	0,5308
	EN - SV	0,545	0,5512	-
XLM-RoBERTa	EN - DE	-	0,6549	0,6559
	EN - NL	0,6355	-	0,6595
	EN - SV	0,6386	0,6677	-
MiniLM	EN - DE	-	0,5684	0,5798
	EN - NL	0,5827	-	0,5842
	EN - SV	0,5714	0,5628	-
MPNET	EN - DE	-	0,5337	0,5676
	EN - NL	0,5921	-	0,5919
	EN - SV	0,58	0,5661	-
DistilSBERT	EN - DE	-	0,5689	0,5845
	EN - NL	0,5807	-	0,5851
	EN - SV	0,5946	0,5696	-