Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра алгоритмических языков

Березникер Алексей Витальевич

# Морфологическая сегментация для маскированных языковых моделей

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**Научный руководитель:**
к.ф-м.н., доцент
Е.И. Большакова

Москва, 2023

# Оглавление

# Аннотация

Данная работа посвящена применению морфологически корректной сегментации слов русского языка для предобученных маскированных языковых моделей BERT. В работе рассматриваются архитектуры современных нейросетевых языковых моделей для решения задач обработки текстов естественного языка и используемые в них алгоритмы токенизации. Исследуется влияние токенизации на качество решения задачи тематической бинарной и многоклассовой классификации. Для русского языка разработана процедура токенизации слов на базе встроенного словаря маскированной языковой модели с учетом морфологически корректной сегментации. Для проведения экспериментов по оценке влияния разработанного способа на качество классификации был построен набор данных, состоящий из редких и морфологически сложных слов, встречающихся в русскоязычных текстах предметных областей: «математика», «биология», «юриспруденция», «экономика». Проведенные эксперименты продемонстрировали, что морфологически корректная сегментация слов может быть использована для улучшения качества работы систем классификации на базе маскированной языковой модели BERT.

# 1 Введение

Автоматическая обработка текстов естественного языка – это обширная область информационных технологий на пересечении машинного обучения и математической лингвистики, связанная с использованием вычислительных средств для анализа текстов естественных языков. К этой области относятся такие задачи, как распознавание речи, выделение смысловых отношений в текстах, кластеризация и категоризация документов, а также реферирование и аннотирование текста [1].

В настоящее время набирает популярность решение прикладных задач обработки текстов естественного языка с использованием языковых моделей. Языковые модели оценивают вероятность различных языковых единиц: символов, токенов (слов или их частей), последовательностей токенов. Основная цель – оценить вероятности фрагментов текста, чтобы эти вероятности отражали знание языка. В частности, предложения, которые с большей вероятностью встречаются в языке, должны иметь большую вероятность.

Языковые модели обучаются на коллекциях текстов – подобранных и обработанных по определенным правилам совокупностям текстов, использующихся в качестве базы для исследования естественного языка и построения систем автоматической обработки текстов. Одним из этапов обработки текстов является токенизация – процесс разбиения текста на элементарные части (слова или подслова), каждая такая часть называется токеном. Основным результатом языковых моделей являются эмбеддинги – векторные представления токенов.

Точно оценить вероятности предложений языка невозможно, поскольку невозможно собрать текстовую коллекцию, содержащую все допустимые предложения на естественном языке. Вместо этого предлагается оценивать вероятность предложения исходя из вероятности его меньших фрагментов [2]. Формально: пусть $y_1, \ldots, y_n$ – токены, $P(y_1, \ldots, y_n)$ – вероятность увидеть все эти токены в тексте в таком порядке. Тогда по правилу цепочки [3] эта вероятность раскладывается на условные вероятности каждого токена с учетом предыдущего контекста:

$$P(y_1, \ldots, y_n) = P(y_1) \cdot P(y_2 \mid y_1) \cdot P(y_3 \mid y_1, y_2) \cdot \ldots \cdot P(y_n \mid y_1, y_2, \ldots, y_{n-1}) = \prod_{t=1}^{n} P(y_t \mid y_{<t})$$

Статистические языковые модели отличаются тем, как они вычисляют и оценивают $P(y_t \mid y_1, \ldots, y_{t-1})$ на основе глобальной статистики, полученной по коллекции текстов, предполагая, что вероятность слова зависит только от фиксированного количества предыдущих слов:

$$P(y_t \mid y_1, \ldots, y_{t-1}) = P(y_t \mid y_{t-n+1}, \ldots, y_{t-1}) = \frac{N(y_{t-n+1}, \ldots, y_{t-1}, y_t)}{N(y_{t-n+1}, \ldots, y_{t-1})},$$

где $N(y_1, \ldots, y_n)$ – количество раз, которое последовательность токенов $y_1, \ldots, y_n$ встречается в коллекции текстов.

В отличие от статистических языковых моделей, нейросетевые языковые модели обучаются предсказывать вероятности $P$. Оценка $P(y_t|y_1, \ldots, y_{t-1})$ сводится к классической задаче классификации: предсказания вероятности встречаемости токена $y_t$ по предшествующим токенам $(y_1, \ldots, y_{t-1})$. Основным результатом нейросетевых языковых моделей являются векторные представления токенов – эмбеддинги. Так, один токен представляется в виде вектора вещественных чисел $\omega = (\omega_1, \ldots, \omega_D) \in \mathbb{R}^D$. На практике в каждой языковой модели есть фиксированный словарь размера $V$ с токенами, в котором содержатся индексы (или указатели), по которым в таблице размера $V \times D$ находятся векторные представления для данных токенов. Наиболее используемыми методами построения векторного представления слов являются Word2Vec [4] и GloVe [5].

В отличие от классической языковой модели, когда оценивается вероятность следующего токена по предыдущим, маскированные языковые модели оценивают вероятность текущего токена по всему входному контексту. Наиболее используемой маскированной языковой моделью является BERT [6], обученная на огромной коллекции текстов английского языка. В результате обучения модели BERT получаются контекстуализированные эмбеддинги, т.е. токены в зависимости от семантики в контексте предложения имеют разные эмбеддинги.

Для обработки текстов в маскированных языковых моделях используются алгоритмы токенизации слов на токены и словари фиксированного размера часто встречающихся токенов. Наиболее используемыми алгоритмами токенизации слов на токены являются BPE [7] и WordPiece [8]. Поскольку редкие и морфологически сложные слова могут отсутствовать в коллекции текстов, на которых обучаются языковые модели, они могут быть не представлены в словаре токенов, что может привести к их некорректной обработке и ухудшению итогового качества решения прикладной задачи. В частности, редкие и морфологически сложные слова могут токенизироваться морфологически некорректно, когда граница разбиения слова проходит не между морфемами – минимальными значимыми единицами текста.

В работе [9] показано, что для английского языка в модели BERT редкие и морфологически сложные слова, как правило, токенизируются морфологически некорректно. Авторами работы был предложен подход к морфологически корректной сегментации слов текста без изменения словаря токенов, использующегося в модели BERT, и экспериментально показано улучшение качества решения задачи тематической бинарной классификации слов английского языка.

В настоящей работе дан обзор архитектур современных нейросетевых маскированных языковых моделей и применяемых в них алгоритмов токенизации. Разработана процедура токенизации слов русского языка на базе встроенного словаря маскированной языковой модели с учетом морфологически корректной сегментации. Для проведения экспериментов по оценке влияния разработанного способа на качество тематической классификации был построен набор данных (датасет), состоящий из редких и морфологически сложных слов, встречающихся в русскоязычных текстах предметных областей: «математика», «биология», «юриспруденция», «экономика». Проведенные эксперименты продемонстрировали значимое улучшение качества классификации слов по метрике ROC AUC.

# 2 Постановка задачи

Цель настоящей работы – исследовать качество интерпретации редких и морфологически сложных слов русского языка в зависимости от способа их токенизации маскированной языковой моделью. Для этого требовалось решить следующие задачи:

1. Изучить архитектуры современных нейросетевых маскированных языковых моделей и применяемые в них алгоритмы токенизации.

2. Построить датасет редких и морфологически сложных слов русского языка в нескольких предметных областях с корректной сегментацией на морфемы.

3. Разработать процедуру токенизации слов русского языка на базе встроенного словаря предобученной маскированной языковой модели с учетом морфологически корректной сегментации.

4. Провести исследовательские эксперименты по оценке качества тематической классификации редких и морфологически сложных слов русского языка на базе предобученных маскированных языковых моделей BERT, с использованием встроенной и морфологически корректной сегментацией.

# 3 Предобученные языковые модели

## 3.1 Архитектуры современных нейросетевых моделей

Наиболее известной задачей в обработке текстов естественного языка является машинный перевод (обычно с одного естественного языка на другой). За последнее десятилетие эта задача решается преимущественно с использованием нейронных сетей.

Формально, в задаче машинного перевода у нас есть входная последовательность слов $x = (x_1, \ldots, x_n)$ и выходная последовательность слов $y = (y_1, \ldots, y_m)$, причем их длина может быть разной ($n \neq m$). Задачу перевода можно рассматривать как нахождение целевой последовательности $y$, которая является наиболее вероятной с учетом входной последовательности $x$, т.е. определяется как

$$y^* = \arg\max_y p(y|x, \theta),$$

где $\theta$ – некоторые параметры алгоритма машинного обучения.

### Нейросетевая архитектура «Sequence-to-sequence»

Нейросетевая архитектура кодировщик-декодировщик (sequence-to-sequence), предложенная в работе [10], используется для решения задач, в которых входные и выходные данные представлены в виде последовательностей. Кодировщик считывает каждый элемент входной последовательности $x$ и переводит полученную информацию в **вектор контекста** $h$. После обработки входной последовательности кодировщик пересылает вектор контекста декодировщику, который начинает генерировать выходную последовательность $y$ элемент за элементом.

Для реализации кодировщика и декодировщика обычно используют рекуррентную нейронную сеть (в частности, LSTM [11] или GRU [12]), которая на каждом этапе принимает на вход два элемента: непосредственно входной элемент и скрытое состояние с предыдущего этапа.

В такой концепции базовым предположением является то, что кодировщик сможет поместить всю информацию о входной последовательности $x$ в вектор контекста $h$, а декодировщик сможет сгенерировать выходную последовательность $y$ на основе этого вектора. Однако наличие единственного вектора контекста приводит к проблеме потери информации: для кодировщика сложно сжать всю необходимую информацию о входной последовательности, а для декодировщика на разных этапах может быть полезна различная информация о входной последовательности.

### Механизм внимания (attention)

Решение этой проблемы было предложено в работе [13], в которой ввели **механизм внимания** – часть нейронной сети, которая на каждом этапе декодировщика решает, какие скрытые состояния кодировщика более важны для генерации текущего элемента выходной последовательности. Основным отличием является то, что кодировщик передает все свои скрытые состояния декодировщику, а декодировщик проходит через дополнительный **слой внимания**, прежде чем сгенерировать очередной элемент выходной последовательности.

Рассмотрим подробнее как работает слой внимания в декодировщике:

1. На каждом этапе имеется предыдущее состояние декодировщика $h_t$ и все состояния кодировщика $s_1, s_2, \ldots, s_n$.

2. Для каждого состояния кодировщика $s_k$ вычисляется $\text{score}(h_t, s_k)$ – «релевантность» для состояния декодировщика $h_t$; на практике «релевантность» вычисляется одним из следующих способов [13, 14]:

   - $\text{score}(h_t, s_k) = h_t^T s_k$ – скалярное произведение векторов $h_t$ и $s_k$
   - $\text{score}(h_t, s_k) = h_t^T W_0 s_k$
   - $\text{score}(h_t, s_k) = w_2 \cdot \tanh(W_1[h_t, s_k])$, где $\tanh(x) = \dfrac{\sinh(x)}{\cosh(x)} = \dfrac{e^{2x} - 1}{e^{2x} + 1}$

3. Вычисляется вектор весов внимания:

$$a = \{\, \text{SoftMax}(\text{score}(h_t, s_k)) \,\}_{k=1}^{n}, \text{ где } a_k = \text{SoftMax}(\text{score}(h_t, s_k)) = \frac{e^{\text{score}(h_t, s_k)}}{\sum\limits_{i=1}^{n} e^{\text{score}(h_t, s_i)}}$$

4. Вычисляется взвешенная сумма состояний кодировщика с весами внимания:

$$c_t = \sum_{k=1}^{n} (a_k \cdot s_k)$$

5. Вычисляется текущее состояние декодировщика:

$$h_{t+1} = f(W[c_t, h_t] + b)$$

Основная идея заключается в том, что сеть может выучить, какие представления входной последовательности более важны на каждом этапе декодировщика. Однако и в такой архитектуре есть важный недостаток: требуется много времени для обучения и применения модели, ведь кодировщик последовательно, элемент за элементом, обрабатывает входную последовательность.

**Нейросетевая архитектура «Трансформер»**

Трансформер – это архитектура нейронной сети, представленная в работе [15], основанная исключительно на механизмах внимания. Архитектура трансформера представила новую парадигму обработки текста: в отличие от предыдущих моделей, где обработка в кодировщике и декодировщике выполнялась с реккурентностью, трансформер работает, используя только механизмы внимания. Преимущество трансформеров применительно к машинному переводу заключается в высоком качестве перевода, а также высокой эффективности за счет перехода от последовательной обработки входного предложения к параллельной.

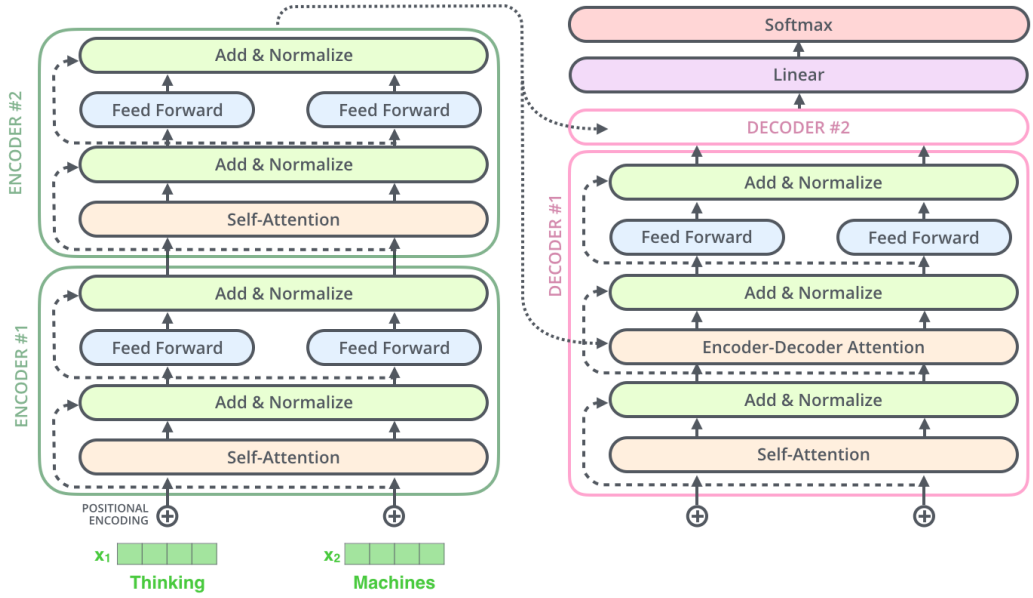



Рисунок 1: Нейросетевая архитектура «Трансформер»

Трансформер состоит из кодирующей компоненты – стека кодировщиков, и декодирующей компоненты – стека декодировщиков, представленных в том же количестве (см. рисунок 1). Все кодировщики идентичны по структуре, но имеют разные веса. Каждый можно разделить на два подслоя. Входная последовательность токенов, поступающая в кодировщик, сначала проходит через слой внутреннего внимания (self-attention), помогающий кодировщику учесть все токены во входной последовательности во время кодирования конкретного токена. Выход слоя внутреннего внимания отправляется в нейронную сеть прямого распространения (feed-forward neural network). Точно такая же сеть независимо применяется для каждого токена в предложении. Декодировщик также содержит два этих слоя, но между ними есть слой внимания, который помогает декодировщику фокусироваться на релевантных выходах кодировщика. Разница между механизмом внимания и механизмом внутреннего внимания заключается в том, что второй работает в рамках одной компоненты модели.

### BERT

BERT [6] – это стек кодировщиков трансформера. В отличие от классической языковой модели, когда оценивается вероятность следующего токена по предыдущим, BERT является маскированной языковой моделью, цель которой оценить вероятность текущего токена по всему входному контексту.

По сравнению с оригинальной архитектурой кодировщика трансформера в модели BERT (см. рисунок 2) модифицирован слой внутреннего внимания и большее число «голов» внимания (multi-head attention), чтобы повысить способность модели фокусироваться на разных частях входной последовательности, ведь одно слово может встречаться в разных контекстах.

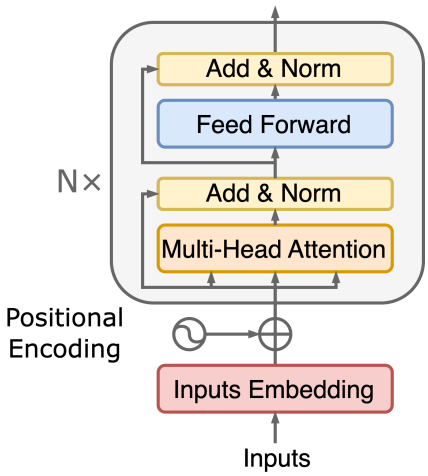



Рисунок 2: Нейросетевая архитектура маскированной языковой модели BERT

Оригинальная модель BERT была обучена на коллекции английских текстов из Wikipedia и BookCorpus одновременно на двух задачах: предсказание токена по контексту с маскированными токенами и определение наличия связи двух последовательных предложений. В первой задаче для каждого входного предложения из обучающей коллекции текстов заранее генерируется 10 разных «масок» по следующему принципу: из входной последовательности каждый токен выбирается с вероятностью $p = 15\%$, затем этот токен в 80% случаев заменяется на специальный токен `[MASK]`, в 10% случаев – на случайный токен, в оставшихся 10% случаев – остается без изменений. Во второй задаче половина обучающих примеров содержат последовательные предложения, извлеченные из обучающей коллекции текстов, а другая половина – случайную пару предложений. Для того, чтобы модель понимала порядок слов, к эмбеддингам входных токенов добавляются векторы кодирования позиций токена (позиционное кодирование).

В результате обучения модели BERT получаются контекстуализированные эмбеддинги токенов, т.е. токены имеют разные вектора на основе их семантики в контексте предложения.

## 3.2 Алгоритмы токенизации

**Токенизация** – процесс разбиения текста на элементарные части, каждая такая часть называется **токеном**. Наиболее распространенными являются алгоритмы токенизации на слова, подслова или символы. На рисунке 3 продемонстрирован возможный результат работы этих алгоритмов применительно к английским словам «learning», «learned» и «deep learning».

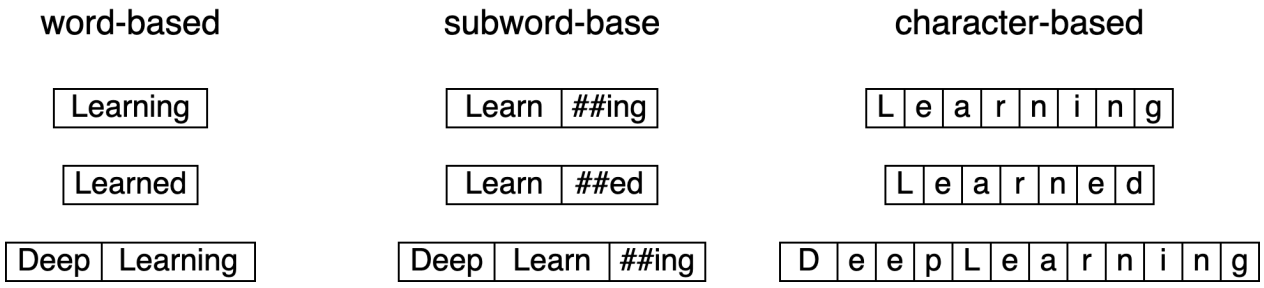



Рисунок 3: Примеры токенизации в разных подходах

Токенизация на основе слов – это самый простой и интуитивный подход к токенизации, он разбивает текст на слова на основе разделителей и эвристических правил. Наиболее часто используемыми разделителями являются пробельные символы и знаки препинания. Такой тип токенизации приводит к словарю большого размера при обработке массивных коллекций текстов, ведь каждое уникальное слово рассматривается как отдельный токен. Это ведет к увеличению размера и сложности языковой модели и требует больших вычислительных ресурсов (памяти и скорости) при обучении и применении модели. Попытки ввести ограничения на размер словаря приводят к потере информации и проблеме out-of-vocabulary (OOV) слов, когда требуемого слова не оказывается в словаре модели.

Токенизация на основе символов – это подход, при котором текст разбивается на отдельные символы. Языки имеют фиксированное количество символов, что приводит к небольшому размеру словаря и решению проблемы OOV. Например, в английских текстах около 256 различных символов: букв, цифр и спецсимволов, в то время как в словарном запасе, согласно Оксфордскому словарю, более 600 тысяч слов. Однако этот подход имеет значительные недостатки, потому что символ обычно не является смысловой единицей и никак не передает окружающий контекст. Кроме того, токенизированная последовательность слов получается намного длиннее исходного текста, ведь каждое слово разбивается на символы, что также влияет на сложность языковой модели.

Токенизация на основе подслов – промежуточное решение между токенизацией на слова и на символы, вобравшее все лучшее от каждого подхода. Основная идея заключается в том, чтобы решить проблему большого размера словаря и длинных последова-

тельностей с менее значимыми отдельными токенами. Важной особенностью алгоритмов токенизации на основе подслов является то, что они не разделяют часто используемые слова на более мелкие подслова и в то же время разделяют редкие слова на более мелкие значимые подслова. Такие алгоритмы обычно используют специальный символ, чтобы указать, какой токен является началом слова, а какой токен – завершением слова. В языковой модели BERT [6], например, в качестве маркера продолжения слова используется символ «##». Токенизация на основе подслов позволяет модели иметь небольшой размер словаря, а также возможность учесть статистически значимые слова. Важным является возможность модели обрабатывать слово, которое она не встречала при обучении, поскольку процесс токенизации может привести к известным подсловам, например, редкое немецкое слово «Abwasserbehandlungsanlange» означает «очистное сооружение» и является конкатенацией соответствующих слов: «abwasser» (сточные воды) + «behandlung» (обработка) + «anlage» (предприятие).

Рассмотрим основные алгоритмы токенизации на подслова.

### Byte Pair Encoding

Изначально Byte Pair Encoding (BPE) был разработан как жадный алгоритм сжатия данных [16], в котором наиболее частотная пара последовательных байтов данных заменяется новым байтом, который не встречается в этих данных. Модификация данного алгоритма [7] нашла свое применение в токенизации на подслова для языковых моделей GPT-2 [17] и RoBERTa [18].

Этапы токенизации заключаются в следующем:

1. Словарь инициализируется всеми уникальными символами из обучающего текста.
2. Текст токенизируется на слова и в конец каждого слова добавляется специальный символ (например, </w>).
3. Вычисляется частота встречаемости слов в тексте.
4. Слова разбиваются на символы (байты) и вычисляется частота символов.
5. Для заданного числа итераций (единственный параметр алгоритма) на каждом шаге вычисляется частота соседних пар байтов, а затем объединяется и добавляется в словарь часто встречающаяся пара байтов.

На практике пары байтов на границе слов отдельно не рассматриваются, что позволяет получившиеся последовательности символов интерпретировать как подслова. С увеличением числа операций слияния размер словаря увеличивается, но количество токенов, используемых для представления текста, уменьшается.

Для токенизации слов, которых не было в обучающей коллекции текстов, сформированные правила слияния будут применяться до тех пор, пока эти слова не включают символы, которых нет в словаре.

Преимущество алгоритма BPE заключается в том, что он может эффективно сбалансировать размер словаря и количество токенов, необходимых для кодирования текста. Алгоритм BPE гарантирует, что наиболее распространенные слова представлены в словаре как один токен, в то время как редкие слова, скорее всего, будут разбиты на два или более токенов. Недостатком является то, что алгоритм BPE основан на жадной и детерминированной замене символов, которая не может обеспечить множественную токенизацию с вероятностями, потому что всегда рассматривается только одно разбиение слова.

### Unigram Language Model

Unigram Language Model (ULM) [19] – это вероятностный алгоритм, который как выбирает пары соседних символов так и принимает решение об их слиянии на каждой итерации на основе вероятности. Предполагается, что каждое подслово $x_i \in \mathcal{V}$ встречается независимо, и, следовательно, вероятность последовательности подслов $x = (x_1, \ldots, x_M)$ формально равна произведению вероятностей подслов:

$$P(x) = \prod_{i=1}^{M} p(x_i),$$

где $\sum\limits_{x \in \mathcal{V}} p(x) = 1$. Наиболее вероятная сегментация $x^*$ для входного слов $x$ задается как

$$x^* = \arg\max_{x \in S(x)} P(x),$$

где $S(x)$ – набор всех возможных сегментаций для слова $x$. Модель подходит к решению проблемы слияния, вычисляя вероятность каждой комбинации подслов, а не выбирая наиболее частотные пары (как в алгоритме BPE). Данный алгоритм используется в языковых моделях XLNet [20] и ALBERT [21].

Этапы токенизации заключаются в следующем:

1. Словарь $\mathcal{V}$ инициализируется большим количеством символов, слов и наиболее распространенных подслов (здесь применима любая эвристика).

2. На каждой итерации алгоритма определяется потеря $\mathcal{L}$:

$$\mathcal{L} = -\sum_{i=1}^{|D|} \log \left( \sum_{x \in S(x_i)} P(x) \right)$$

3. Максимизируется $P(x)$ с помощью EM-алгоритма [22].

4. Для каждого элемента в словаре вычисляется, насколько увеличится общая потеря $\mathcal{L}$, если элемент будет удален из словаря.

5. Удаляется p% (параметр алгоритма, обычно 10%) элементов словаря, для которых увеличение потери является наименьшим, т.е. те подслова, которые меньше всего влияют на потерю $\mathcal{L}$. Однако алгоритм всегда сохраняет одиночные символы, чтобы любое слово могло быть токенизировано.

6. Процесс повторяется до тех пор, пока словарный запас не достигнет заданного размера (параметр алгоритма).

Поскольку алгоритм Unigram не основан на правилах слияния (в отличие от алгоритма BPE), существует несколько способов токенизации нового слова. Однако данный алгоритм сохраняет вероятность каждого токена из словаря, так что вероятность каждой возможной токенизации слова может быть вычислена после обучения. На практике алгоритм Unigram просто выбирает наиболее вероятную токенизацию для каждого слова.

**WordPiece**

WordPiece [8] – это фактически алгоритм BPE (хотя формально WordPiece появился раньше), в котором при слиянии токенов максимизируется правдоподобие $\mathcal{L}$. Единственное различие между алгоритмами WordPiece и BPE – это способ добавления пар токенов в словарь. WordPiece – это жадный алгоритм, который использует вероятность вместо частоты, чтобы объединить лучшую пару соседних токенов на каждой итерации, однако выбор токена для пары все еще основан на частоте встречаемости пары. Таким образом, алгоритм WordPiece похож на алгоритм BPE с точки зрения выбора токена для пары и похож на алгоритм Unigram с точки зрения выбора лучшей пары для слияния. Данный алгоритм используется в языковой модели BERT [6].

Этапы токенизации заключаются в следующем:

1. Словарь инициализируется уникальными символами.

2. Строится языковая модель на обучающих данных, используя словарь из 1 этапа.

3. Создается новый токен, являющийся объединением двух токенов из текущего словаря, так, чтобы он меньше других пар увеличивал потерю $\mathcal{L}$.

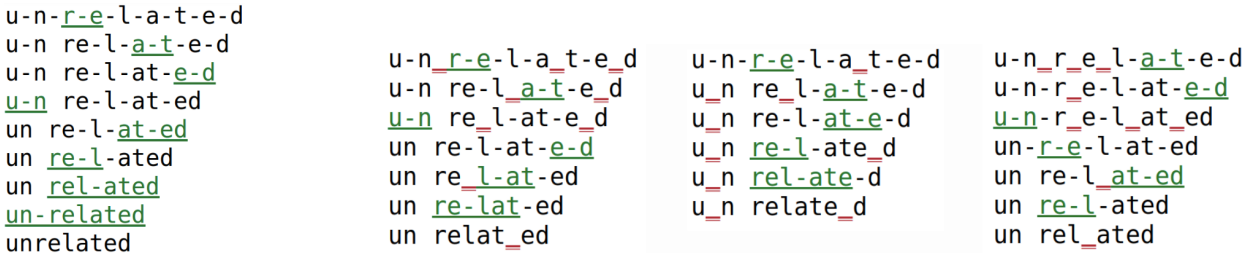
4. Процесс повторяется до тех пор, пока не будет достигнут заранее определенный размер словаря (параметр алгоритма) или уменьшение вероятности не упадет ниже определенного порога (параметр алгоритма).

### BPE-Dropout

BPE-Dropout [23] – алгоритм токенизации на подслова, разработанный в компании Яндекс, являющийся модификацией алгоритма BPE и показывающий на практике результат лучше алгоритмов BPE и Unigram.

Недостаток алгоритма BPE в его детерминированности: он разбивает слова на уникальные последовательности подслов, а это означает, что для каждого слова модель учитывает только одно разбиение. Таким образом, модель, скорее всего, не достигнет своего полного потенциала в учете морфологии слов и устойчивости к ошибкам токенизации. Для борьбы с этим недостатком предлагается использовать словарь и таблицу слияний алгоритма BPE, но на каждом шаге слияния с вероятностью $p$ его пропускать. Если $p = 0$, токенизация эквивалентна стандартному алгоритму BPE, если $p = 1$, алгоритм разбивает слова на отдельные символы. Значения от 0 до 1 можно использовать для управления детализацией токенизации. Во время обучения используется $p > 0$ (обычно 0.1), а во время применения модели используется $p = 0$.

На рисунке 4 продемонстрирован процесс токенизации английского слова «unrelated» (несвязанный) с использованием алгоритмов BPE (a) и BPE-Dropout (b). Дефисы указывают на возможные слияния, которые присутствуют в таблице слияний, зеленым цветом отображены слияния, выполняемые на каждом шаге, красным – пропущенные слияние. Таким образом, при обучении языковая модель с алгоритмом токенизации BPE-Dropout для слова «unrelated» вместо одной токенизации «un related» сможет увидеть такие варианты токенизации, как «un relat ed», «u n relate d», «un rel ated» и другие.




(a) BPE (b) BPE-Dropout

Рисунок 4: Процесс токенизации английского слова «unrelated» с использованием: (a) BPE, (b) BPE-Dropout

## 3.3 DelBERT: BERT с морфологически корректной сегментацией слов

Работа [9] посвящена исследованию вопроса, как сегментация текста для предварительно обученных языковых моделей влияет на их интерпретацию редких и морфологически сложных слов. Показывается, что в модели BERT для английского языка с сегментацией встроенным алгоритмом WordPiece редкие слова часто разбиваются морфологически некорректно, т.е. граница разбиения слова на подслова может проходить не между морфемами. В качестве морфем рассматриваются основы (stem) и аффиксы (affix): префиксы и суффиксы. Часто встречающиеся в текстах английского языка слова, например, «stabilize» (стабилизировать) и «realize» (осознавать), полностью хранятся в словаре языка модели BERT как токены, а редкие слова разбиваются на подслова, при этом более редкие слова, например, «mobilize» (мобилизовать) и «templatize» (шаблонизировать), разбиваются морфологически некорректно (см. таблицу 1).

Таблица 1: Примеры сегментации алгоритмом WordPiece в модели BERT

| Слово | Токены |
|---|---|
| stabilize | [ stabilize ] |
| realize | [ realize ] |
| finalize | [ final, ##ize ] |
| mobilize | [ mob, ##il, ##ize ] |
| tribalize | [ tribal, ##ize ] |
| templatize | [ te, ##mp, ##lat, ##ize ] |

Для решения проблемы морфологически некорректной сегментации в рассматриваемой работе [9] был взят алгоритм сегментации из работы [24], который для заданного набора аффиксов и основ английского языка производит морфологическую сегментацию слова. Ниже представлено описание работы этого алгоритма на псевдокоде:

$A \leftarrow$ набор словообразовательных аффиксов для английского языка;
$S \leftarrow$ набор основ английского языка;
$B_1^A(w) \leftarrow$ множество слов $w$,
образованное удалением одного из словообразовательных аффиксов из $A$;
**while** $B_{i+1}^A(w) \cap S \neq \varnothing$ **do**
  $B_{i+1}^A(w) = \cup_{b \in B_i^A(w)} B_1^A(b)$;
**end**

Приведем пример работы этого алгоритма для английского слова «unlockable» (разблокируемый):

$B_1^A(\text{unlockable}) = \{\text{lockable}, \text{unlock}\}$

$B_2^A(\text{unlockable}) = \{\text{lock}\}$

$\varnothing \Rightarrow$ «unlockable» = [ un, ##lock, ##able ]

Описанный алгоритм учитывает большинство морфо-орфографических особенностей английского языка. Например, когда суффикс «ize» удаляется из слова «isotopize», результирующим словом будет «isotope», а не «isotop».

В работе [9] в качестве входных данных $A$ и $S$ для рассмотренного алгоритма использовались аффиксы и основы из словаря модели BERT, построенного алгоритмом WordPiece. Это означает, что все токены, используемые в рассмотренном алгоритме сегментации, также доступны для сегментации алгоритмом WordPiece, и результирующую сегментацию слов можно применять в предобученной языковой модели BERT.

В экспериментах, проведенных в работе [9], сравнивалось качество классификации модели BERT с разными алгоритмами $s(x)$ сегментации слов $x$. Проверялось, насколько $s(x)$ влияет на способность модели BERT предсказать тематический класс $y$ слова $x$. Рассматривалась классическая постановка задачи бинарной классификации:

- Пусть $x$ – редкое/сложное слово, $y$ – метка класса.
- $s(x) = (t_1, \ldots, t_k)$ – сегментация слова $x$ на последовательность из $k \geqslant 1$ токенов.
- Тогда $P(y|s(x))$ – вероятность истинного тематический класса $y$ при заданной сегментации $s(x)$.

В качестве источников редких слов в работе [9] использовались три больших набора данных на английском языке:

1. Amazon – набор данных с отзывами и оценками, полученным на коммерческой онлайн-платформе Amazon. Для отзывов с 4 или 5 звездами устанавливался положительный класс (pos), а для отзывов с 1 или 2 звездами – отрицательный класс (neg), отзывы с 3 звездами не рассматривались.

2. ArXiv – тексты статей с электронного архива научных статей ArXiv на тему «physics» (phy) и «computer science» (cs). Были выбраны именно эти темы, поскольку для них тематическое расстояние больше по сравнению с другими парами, такими как «mathematics» и «computer science».

3. Reddit – набор данных, состоящий из текстов обсуждений на темы «entertainment» (ent): anime, DestinyTheGame, funny, Games, gaming, leagueoflegends, movies, Music, pics, videos; и «discussion» (dis): skscience, atheism, conspiracy, news, Libertarian, politics, science, technology, TwoXChromosomes, worldnew.

Примеры редких и морфологически сложных слов, отобранных из этих наборов данных приведены в таблице 2.

Таблица 2: Примеры редких слов в наборах данных

| Набор данных | # слов | Примеры редких слов |
|---|---|---|
| Amazon | 239 727 | overpriced, megafavorite |
| ArXiv | 97 410 | semithermal, rankable |
| Reddit | 85 362 | supervampires, immigrationism |

Были обучены две модели для задачи бинарной классификации на базе BERT со стандартной сегментацией $s_w(x)$ алгоритмом WordPiece и с использованием морфологически корректного алгоритма сегментации $s_d(x)$. Вторую модель назвали DelBERT (**De**rivation **l**everaging **BERT**). Модели отличаются только способом сегментации рассматриваемых редких и морфологически сложных слов.

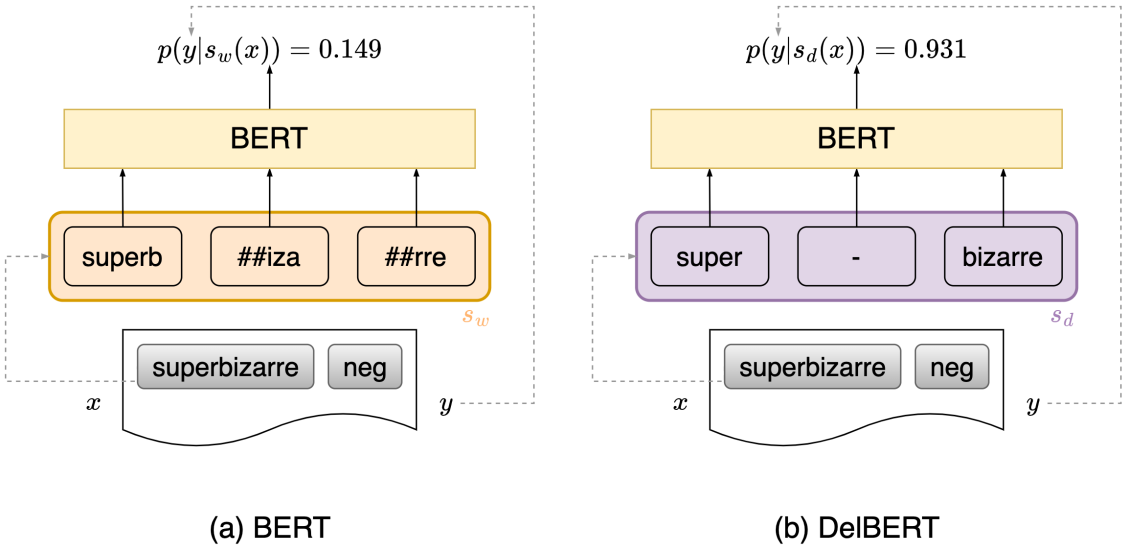



Рисунок 5: Классификация слова «superbizarre» моделью (a) BERT и (b) DelBERT

В экспериментах было продемонстрировано, что модель BERT с алгоритмом морфологически корректной сегментации $s_d(x)$ уверенно предсказывает правильный тематический класс для отобранных редких и морфологически сложных слов, в то время как модель BERT с сегментацией $s_w(x)$ алгоритмом WordPiece ошибается. В частности, для редкого слова «superbizarre» (см. рисунок 5).

В таблице 3 продемонстрированы успешные примеры морфологически корректной сегментации и правильной классификации слов для модели DelBERT, где $x$ – слово, $y$ – класс, $s_w(x)$ и $s_d(x)$ – алгоритмы сегментации, $\mu_p = p(y|s_*(x))$ – предсказание модели.

Таблица 3: Примеры сегментации редких слов
и их классификации моделями BERT и DelBERT

| Набор данных | $x$ | $y$ | $s_d(x)$ | $\mu_p$ | $s_w(x)$ | $\mu_p$ |
|---|---|---|---|---|---|---|
| Amazon | applausive | pos | applause, ##ive | 0.847 | app, ##laus, ##ive | 0.029 |
| | superannoying | neg | super, -, annoying | 0.967 | super, ##ann, ##oy, ##ing | 0.278 |
| | overseasoned | neg | over, -, seasoned | 0.956 | overseas, ##oned | 0.219 |
| ArXiv | isotopize | phy | isotope, ##ize | 0.985 | iso, ##top, ##ize | 0.039 |
| | antimicrosoft | cs | anti, -, microsoft | 0.936 | anti, ##mic, ##ros, ##oft | 0.013 |
| | inkinetic | phy | in, -, kinetic | 0.983 | ink, ##ine, ##tic | 0.035 |
| Reddit | prematuration | dis | premature, ##ation | 0.848 | prem, ##at, ##uration | 0.089 |
| | nonmultiplayer | ent | non, -, multiplayer | 0.950 | non, ##mu, ##lt, ##ip, ##layer | 0.216 |
| | promosque | dis | pro, -, mosque | 0.961 | promo, ##sque | 0.066 |

В экспериментах качество классификации измерялось $F_1$-мерой (среднее гармоническое значение точности и полноты) на тренировочном (Train) и тестовом (Test) наборах данных. Как видно из таблицы 4, для всех наборов слов модель DelBERT показывает качество, которое статистически значимо выше качества модели BERT со встроенным алгоритмом сегментации WordPiece.

Таблица 4: DelBERT: качество классификации $F_1$-мера

| Модель | Amazon | | ArXiv | | Reddit | |
|---|---|---|---|---|---|---|
| | Train | Test | Train | Test | Train | Test |
| **DelBERT** | $.635 \pm .001$ | $.639 \pm .002$ | $.731 \pm .001$ | $.723 \pm .001$ | $.696 \pm .001$ | $.701 \pm .001$ |
| BERT | $.619 \pm .001$ | $.624 \pm .001$ | $.704 \pm .001$ | $.700 \pm .002$ | $.664 \pm .001$ | $.664 \pm .003$ |

Таким образом, в работе [9] было показано, что для английского языка в языковой модели BERT редкие и морфологически сложные слова, как правило, сегментируются морфологически некорректно, что отражается на качестве тематический классификации, но качество может быть повышено за счет морфологически корректной сегментации. В то же время, аналогичный вопрос о влиянии способа сегментации на качество тематической классификации слов русского языка маскированными языковыми моделями оставался открытым.

# 4 Модель DelRuBERT

## 4.1 Общая характеристика модели

**DelRuBERT** – разработанная в настоящей работе маскированная языковая модель для русского языка, использующая предобученную модель RuBERT и морфологически корректную сегментацию слов, при которой граница разбиения слов проходит между морфемами – минимальными значимыми единицами текста. Важно, что для токенизации используются только те токены, которые есть в словаре предобученной модели RuBERT. Это означает, что все токены, используемые при токенизации с учетом морфологической сегментации, также доступны для токенизации встроенным алгоритмом предобученной маскированной языковой модели RuBERT.

Для построения и обучения модели DelRuBERT, а также проведения экспериментов по тематической классификации было необходимо:

1. Выбрать предобученные языковые модели RuBERT для русского языка.
2. Уточнить решаемую задачу классификации и построить датасет редких и морфологически сложных слов русского языка.
3. Разработать процедуру токенизации слов русского языка для выбранных языковых моделей RuBERT с учетом морфологически корректной сегментации.
4. Провести исследовательские эксперименты по сравнению качества классификации модели RuBERT для встроенной и для морфологически корректной сегментацией.

Существуют две классические архитектуры маскированной языковой модели BERT: Base и Large. Также есть cased и uncased виды этих моделей, которые отличаются тем, учитывается ли регистр символов (cased) или нет (uncased). Основные отличия между видами оригинальной модели BERT [6] показаны в таблице 5. Отметим, что размер словаря токенов у моделей BERT Base и BERT Large совпадает.

Таблица 5: Сравнение видов оригинальной модели BERT

| BERT | Base cased | Base uncased | Large cased | Large uncased |
|---|---|---|---|---|
| Количество слоев кодировщика | 12 | 12 | 24 | 24 |
| Количество «голов» внимания | 12 | 12 | 16 | 16 |
| Размерность скрытых слоев | 768 | 768 | 1024 | 1024 |
| Количество параметров | 110 M | 110 M | 340 M | 340 M |
| Размер словаря | 28996 | 30522 | 28996 | 30522 |

В качестве предобученных языковых моделей были выбраны наиболее используемые (по данным с сайта huggingface.co) модели BERT Base, обученные на коллекции русских текстов: Sber RuBERT и DeepPavlov RuBERT. Обе модели имеют словарь размером ≈120 тысяч токенов. При этом словари моделей на 41% состоят из непересекающегося множества токенов, поскольку используются разные алгоритмы токенизации.

- **DeepPavlov RuBERT** [25] – мультиязычная модель BERT Base (uncased), дообученная научно-исследовательской командой DeepPavlov на русскоязычной части Википедии и новостных данных и опубликованная в открытый доступ весной 2020 года. В качестве алгоритма токенизации используется WordPiece. Размер словаря – 119547 токенов.

- **Sber RuBERT** [26] – мультиязычная модель BERT Base (uncased), дообученная разработчиками кампании SberDevices совместно с коллегами из DeepPavlov и опубликованная в открытый доступ летом 2021 года. Модель обучалась на коллекции русских текстов, в которой была Википедия, новости, часть корпуса Taiga [27] и небольшой массив книг. В качестве алгоритма токенизации используется BPE. Размер словаря – 120138 токенов.

Оригинальные модели BERT не рассматривались в настоящем исследовании, поскольку в их словарях было значительно меньше токенов, содержащих кириллические символы, чем в словарях моделей Sber RuBERT и DeepPavlov RuBERT (см. таблицу 6).

Таблица 6: Процент токенов словаря, содержащих кириллические символы в разных моделях BERT

| Модель BERT | % токенов с кириллицей |
|---|---|
| BERT Base cased | 0.28 |
| BERT Base uncased | 0.48 |
| BERT Large cased | 0.28 |
| BERT Large uncased | 0.48 |
| BERT Base uncased (мультиязычная) | 11.55 |
| **DeepPavlov RuBERT** | 83.34 |
| **Sber RuBERT** | 94.04 |

Поскольку исследуется влияние способа токенизации маскированной языковой модели на качество тематический классификации слов русского языка, необходимо было определить классификатор над эмбеддингами модели RuBERT. В качестве классификатора рассматривались простые полносвязные нейронные сети: однослойная и двухслойная с разными функциями активации между линейными слоями:

- $\mathrm{ReLU}(x) = x \cdot \mathbb{I}[x > 0]$, где $\mathbb{I}$ – функция-индикатор проверки истинности условия.
- $\mathrm{LeakyReLU}(x) = x \cdot \mathbb{I}[x > 0] + \alpha x \cdot \mathbb{I}[x \leqslant 0]$
- $\mathrm{GeLU}(x) = x \cdot \Phi(x)$, где $\Phi(x)$ – функция нормального распределения.

Однослойная нейронная суть по сути представляет логистическую регрессию. Двухслойная сеть может оказаться слишком сложной для решения задачи классификации, поэтому для минимизации риска переобучения и роста обобщающей способности классификатора между линейными слоями был добавлен слой Dropout [28], который с вероятностью $p$ исключает нейрон при обучении модели – это означает, что при любых входных данных нейрон возвращает 0. Размер последнего полносвязного слоя классификатора с функцией активации SoftMax задает количество тематических классов, что позволяет проводить эксперименты как с бинарной, так и с многоклассовой классификацией.

Архитектура модели DelRuBERT представлена на рисунке 6.

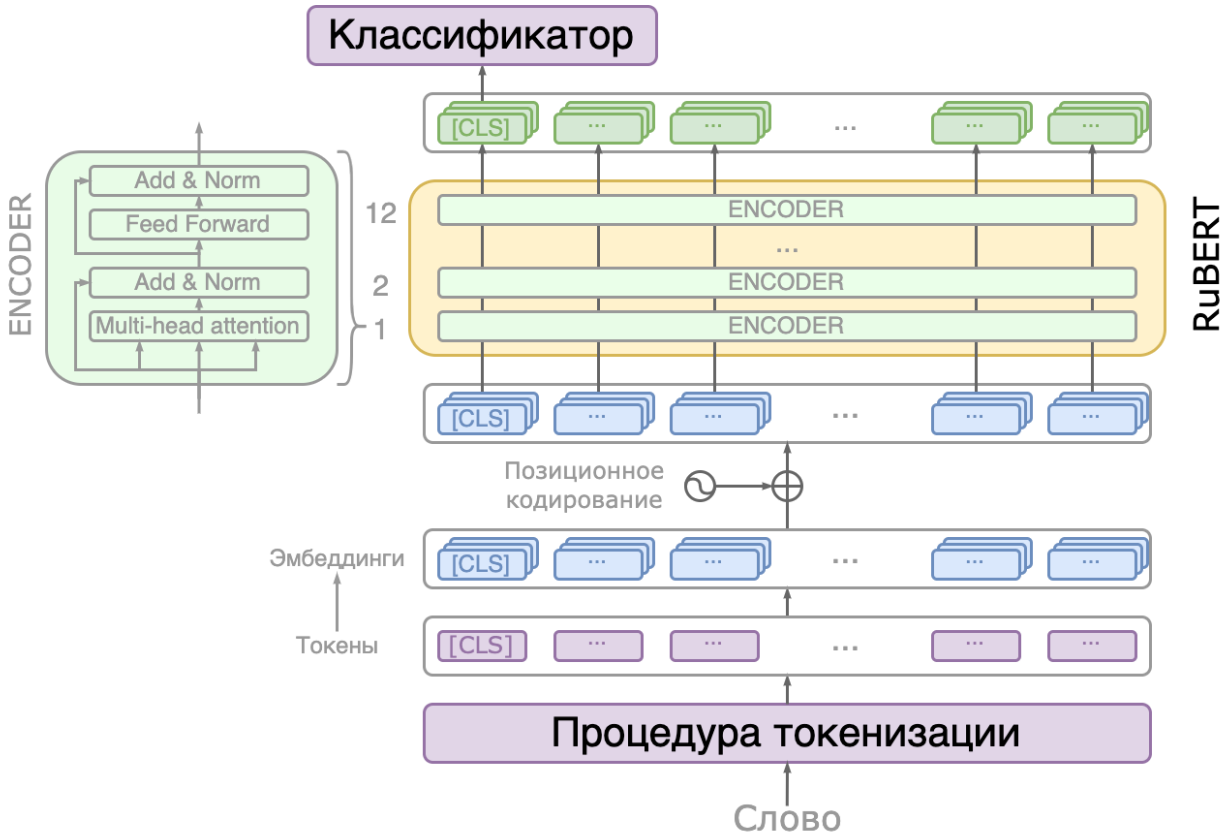



Рисунок 6: Архитектура модели DelRuBERT

Реализация нейросетевого классификатора описанной архитектуры написана на языке программирования Python 3 с использованием фреймворка машинного обучения

PyTorch [29]. Для получения морфологически корректной сегментации слов использовался инструмент автоматического морфемного разбора – CrossMorphy [30].

## 4.2 Построение набора данных для экспериментов

Для проведения экспериментов был построен датасет редких и морфологически сложных слов из областей «математика», «биология», «юриспруденция», «экономика», включающий данные об их морфемной структуре. Построенный датасет получил название – RuRareWords.

Этапы построения датасета RuRareWords заключались в следующем:

1. Сбор коллекции русскоязычных текстов из областей и выбор редких и морфологических сложных слов из собранных коллекций.

2. Морфологически корректная сегментация редких и морфологических слов, отобранных на первом этапе, с использованием инструмента автоматического морфемного разбора – CrossMorphy [30].

3. Конвертация данных в формат json-файла.

На первом этапе построения датасета RuRareWords было необходимо выбрать редкие и морфологически сложные слова, для этого:

- Вручную были собраны коллекции текстов учебной литературы, содержащие несколько сотен тысяч уникальных лемм слов, и словари терминов для каждой из тематических областей.

- В рамках тематической области для каждого слова автоматизировано подсчитана частота $\nu_1$ встречаемости в исходной коллекции текстов и частота $\nu_2$ встречаемости в русскоязычной части Википедии. Это было необходимо для того, чтобы учесть и исключить слова, которые часто встречаются в русскоязычной части Википедии, а значит, и в обучающей коллекции текстов маскированных языковых моделей RuBERT. Слова, для которых $\nu_1 < 3$, не рассматривались, чтобы исключить слова с опечатками.

- Слова упорядочивались по произведению частот встречаемости $\nu_1 \cdot \nu_2$ (от меньшего значения к большему), после чего дополнительно просматривались вручную, чтобы исключить слова, не относящиеся к соответствующей тематической области, имена собственные, а также слова с опечатками (для которых $\nu_1 \geqslant 3$).

В таблице 7 продемонстрированы примеры редких и морфологически сложных слов и их общее количество в каждой из рассматриваемых областей датасета RuRareWords. Отметим, что тематические классы сбалансированы по количеству слов.

Таблица 7: Примеры слов из датасета RuRareWords

| Область | Примеры слов из датасета RuRareWords | # слов |
|---|---|---|
| Математика | аксонометрия, аппроксимация, гиперпирамида, гиперэллипс, директриса, компланарность, мультипликативность, неортогональность, ромбогексаэдр, симплекс, шестидесятиугольник, эвольвента, эквидистантность, эксцентричный, аддитивность | 1356 |
| Биология | вирусоносительство, диссимиляция, интерферон, микроэволюция, монофилия, мутагенез, пролиферация, реверсия, резистентность, синапсис, тотипотентность, фитогормон, фосфопротеиды, фоторецепторы, хроматофор | 1350 |
| Юриспруденция | абдикация, аджастер, гратификация, декриминализация, индоссамент, интернирование, лукративный, менажировать, панаширование, регрессант, сецессия, стипуляция, федерирование, цивилистика, энциклика | 1324 |
| Экономика | авуары, аквизиция, аккредитив, девальвация, диверсификация, дефляция, индоссамент, инкассо, консигнация, концессия, неокейнсианство, олигополия, оферент, паритет, пауперизм, пролонгация, ревальвация, рестрикция, форфейтинг, эмиссия | 1330 |

На втором этапе построения датасета RuRareWords все слова были разбиты на морфемы. Для получения морфологически корректной сегментации слов на токены использовался инструмент автоматического морфемного разбора – CrossMorphy [30] (морфологический анализатор), который в частности выполняет сегментацию слов на морфемы с классификацией по типам (корень, префикс, суффикс, окончание) словарных и не словарных слов [31]. Дополнительно в датасет RuRareWords были добавлены варианты токенизации слов встроенными алгоритмами токенизации, используемыми в выбранных предобученных языковых моделях RuBERT (BPE и WordPiece), в формате со специальным символом «##».

Ниже представлен пример работы морфологического анализатора CrossMorphy на некоторых словах из датасета RuRareWords:

```
$ echo "аксонометрия аддитивность микроэволюция фитогормон федерирование \
рестрикция" | ./xmorphy -m
```

```
+---------------+---------------+-------------+--------+-----------------------------------------+
|      Form     |  Normal form  | Speech Part | Source |              Morphemic parse            |
+---------------+---------------+-------------+--------+-----------------------------------------+
| аксонометрия  | аксонометрия  | NOUN        | DICT   | аксон:ROOT/о:LINK/метр:ROOT/и:SUFF/я:END  |
+---------------+---------------+-------------+--------+-----------------------------------------+
| аддитивность  | аддитивность  | NOUN        | DICT   | аддит:ROOT/ивн:SUFF/ост:SUFF/ь:END        |
+---------------+---------------+-------------+--------+-----------------------------------------+
| микроэволюция | микроэволюция | NOUN        | PREF   | микро:PREF/эволюци:ROOT/я:END             |
+---------------+---------------+-------------+--------+-----------------------------------------+
| фитогормон    | фитогормон    | NOUN        | DICT   | фит:ROOT/о:LINK/гормон:ROOT               |
+---------------+---------------+-------------+--------+-----------------------------------------+
| федерирование | федерирование | NOUN        | SUFF   | федер:ROOT/ир:SUFF/ова:SUFF/ни:SUFF/е:END |
+---------------+---------------+-------------+--------+-----------------------------------------+
| рестрикция    | рестрикция    | NOUN        | DICT   | рестрикц:ROOT/и:SUFF/я:END                |
+---------------+---------------+-------------+--------+-----------------------------------------+
```

В таблице 8 показаны примеры сегментации слов (1 столбец) из датасета RuRareWords для тематик «математика» и «биология» разными алгоритмами сегментации:

- $s_{\text{CrossMorphy}}$ (2 столбец) – морфологически корректная сегментация.
- $s_{\text{SberRuBERT}}$ (3 столбец) – BPE-сегментация, модель Sber RuBERT.
- $s_{\text{DeepPavlovRuBERT}}$ (4 столбец) – WordPiece-сегментация, модель DeepPavlov RuBERT.

Видно, что сегментация слов существенно различается, часто в отдельный токен не выделяется корень – носитель основного смысла слова.

Таблица 8: Примеры сегментации слов из датасета RuRareWords

| | Слово | $s_{\text{CrossMorphy}}(x)$ | $s_{\text{SberRuBERT}}(x)$ | $s_{\text{DeepPavlovRuBERT}}(x)$ |
|---|---|---|---|---|
| Математика | аддитивность | аддит, ##ивн, ##ост, ##ь | ад, ##дит, ##ив, ##ность | аддитив, ##ность |
| | гиперграньь | гипер, грань | гипер, ##гран, ##ь | гиперг, ##рань |
| | дельтоид | дельт, ##оид | дель, ##тои, ##д | дель, ##то, ##ид |
| | изометрия | изо, метр, ##и, ##я | изо, ##метрия | изом, ##етрия |
| | планарный | планар, ##н, ##ый | плана, ##рн, ##ы, ##й | плана, ##р, ##ный |
| Биология | авитаминоз | авитамин, ##оз | ави, ##тами, ##но, ##з | ави, ##тами, ##ноз |
| | биоценоз | био, цен, ##оз | био, ##цено, ##з | био, ##цен, ##оз |
| | гетерогамия | гетер, ##о, гам, ##и, ##я | гетеро, ##гами, ##я | гетеро, ##гами, ##я |
| | идиоплазма | идио, плазм, ##а | идио, ##пла, ##зма | иди, ##оплаз, ##ма |
| | терморегуляция | терм, ##о, регул, ##яци, ##я | термо, ре, гуля, ция | термо, регу, ляция |

На третьем этапе построения для удобства анализа и проведения экспериментов датасет RuRareWords был представлен в виде файла в формате json для каждой из областей, элементами которого являются слова, содержащие лемму и варианты сегментации: морфологически корректную и встроенными алгоритмами токенизации выбранных предобученных маскированных языковых моделей RuBERT. Ниже приведен пример элемента датасета RuRareWords для слова «резистентность» из области «биология»:


```
{
    "lemma": "резистентность",
    "xmorphy": {
        "phem_info": "резистент:ROOT/н:SUFF/ост:SUFF/ь:END",
        "tokens": ["резистент", "##н", "##ост", "##ь" ]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["рези", "##стент", "##ность"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["резист", "##ентность"]
    }
}
```


В Приложении А представлено ещё несколько дополнительных примеров слов для каждой из областей датасета RuRareWords.

После построения датасета RuRareWords был произведен его анализ. Построены распределения длин слов в символах и морфемах для каждой тематической области (см. Приложение Б). Все распределения, за исключением области «экономика», имеют нормальную форму. Наличие слов маленькой длины обосновывается тем, что датасет содержит не только морфологически сложные, но и редкие (низкочастотные) слова. Основные статистики по распределению длин слов представлены в таблице 9.

Таблица 9: Статистика длин слов в тематических областях

| Область | Средняя длина в символах | Медианная длина в символах | Средняя длина в морфемах | Медианная длина в морфемах |
|---|---|---|---|---|
| Математика | 10.4 | 10 | 3.1 | 3 |
| Биология | 9.3 | 9 | 2.9 | 3 |
| Юриспруденция | 8.3 | 8 | 2.6 | 2 |
| Экономика | 8.7 | 8 | 2.5 | 2 |

Реализация этапов построения датасета RuRareWords написана на языке программирования Python 3 и Bash. Код построения и сам датасет доступен по ссылке в git-репозитории: Berezniker/DelRuBERT.

## 4.3 Процедура токенизации слов

Для проведения экспериментов по тематической классификации слов было необходимо разработать процедуру токенизации с учетом морфологически корректной сегментации слов русского языка для выбранных языковых моделей RuBERT. Поскольку для применения предобученной языковой модели допустимо использовать только токены из словаря модели, а сам словарь рассматриваемых языковых моделей ограничен, то при морфологически корректном разбиении слова на морфемы может образоваться токен (морфема), которого в рассматриваемом словаре нет. Поэтому полностью морфологически корректное разбиение слова провести для всех случаев невозможно.

В частности, для слова «бактериоцид» морфологически корректным разбиением является [бактери, ##о, ##цид] – все 3 морфемы данного разбиения присутствуют как токены в словаре предобученной языковой модели Sber RuBERT, а значит, можно произвести морфологически корректную сегментацию слова «бактериоцид» с использованием встроенного словаря модели Sber RuBERT. В то же время токен «##цид» отсутствует в словаре предобученной языковой модели DeepPavlov RuBERT, а значит, нельзя провести полностью морфологически корректную сегментацию слова «бактериоцид» с использованием встроенного словаря модели DeepPavlov RuBERT.

Наш анализ датасета RuRareWords показал, что для словаря модели Sber RuBERT полностью морфологически корректное разбиение слов из датасета RuRareWords можно произвести в 5% случаев для области «юриспруденция», 6% – для области «биология», 7% – для областей «математика» и «экономика», а для словаря модели DeepPavlov RuBERT – в 5% случаев для областей «биология» и «юриспруденция» и в 6% случаев для областей «математика» и «экономика».

Поэтому для морфем, которые не хранятся как отдельные токены в словаре предобученной языковой модели, предлагается проводить дополнительную сегментацию встроенным алгоритмом токенизации предобученной языковой модели.

В таблице 10 представлены результаты экспериментов с оценкой качества бинарной классификации на тестовом наборе данных для тематических классов «математика» и «биология», проведенных с целью уточнения возможности токенизации для корней и аффиксов слова. В качестве классификатора рассматривалась логистическая регрессия. Как видно из таблицы 10 (см. столбец 3) при дополнительной токенизации всех морфем качество классификации значимо не изменяется относительно токенизации встроенным алгоритмом языковой модели (см. столбец 2). Аффиксы в русском языке служат для словообразования новых форм слова, поэтому при их «дроблении» модели становится сложнее выучить данную связь. В то же время подход, при котором дополнительная токенизация применяется только к корням слов, даёт значимое улучшение качества классификации (см. столбец 4).

Таблица 10: Оценка качества разных подходов к токенизации морфем

| Языковая модель | $AUC_{test}$ | | |
|---|---|---|---|
| | встроенная токенизация | токенизация всех морфем | токенизация только корней |
| Sber RuBERT | 0.785 | 0.788 | **0.811** |
| DeepPavlov RuBERT | 0.786 | 0.780 | **0.797** |

Исходя из результатов экспериментов, было принято решение проводить дополнительную токенизация только для морфем, являющихся корнем слова, а аффиксы и окончания не «дробить» на более мелкие единицы и в случае, если аффикса не оказалось в словаре модели, использовать для исходного слова токенизацию встроенным алгоритмом токенизации предобученной языковой модели. Для чистоты проводимых экспериментов такие слова были оставлены в датасете.

К примеру, слово «резистентность» имеет морфологически корректное разбиение [резистент, ##н, ##ост, ##ь]. Морфемы «н», «ост» и «ь» включены как отдельные токены в словари предобученных языковых моделей Sber RuBERT и DeepPavlov RuBERT. Однако корня «резистент» как отдельного токена в словарях этих моделей нет, поэтому оно сегментируется встроенными алгоритмами токенизации на [рези, ##стент] или [резист, ##ент] для моделей Sber RuBERT и DeepPavlov RuBERT соответственно. Таким образом, слово «резистентность» токенизируется как [рези, ##стент, ##н, ##ост, ##ь] для модели Sber RuBERT и как [резист, ##ент, ##н, ##ост, ##ь] для модели DeepPavlov RuBERT.

Блок-схема разработанной процедуры токенизации слов представлена на рисунке 7. Основные этапы заключаются в следующем:

1. На вход процедуре поступает слово $\omega$ и словарь $\Omega$ токенов предобученной языковой модели.

2. Из датасета RuRareWords извлекается морфологически корректная сегментация слова $\omega$ на морфемы $\{\tau_i\}$.

3. В цикле для каждой морфемы $\tau_i$ проверяется ее наличие в словаре $\Omega$ предобученной языковой модели, после чего:

   - Если морфема $\tau_i$ оказалась в словаре $\Omega$, то она в таком виде переходит в итоговую сегментацию.
   - Если морфемы $\tau_i$ не оказалось в словаре $\Omega$ и она является корнем исходного слова, производится токенизация $\tau_i$ встроенным алгоритмом токенизации пре-

добученной языковой модели, результирующие токены переходят в итоговую сегментацию.

- Если морфемы $\tau_i$ не оказалось в словаре $\Omega$ и она не является корнем исходного слова, то для всего исходного слова $\omega$ используется встроенный алгоритм токенизации и это является итоговой токенизация слова $\omega$. Отметим, что таких слов в датасете менее 9% для каждой из областей.

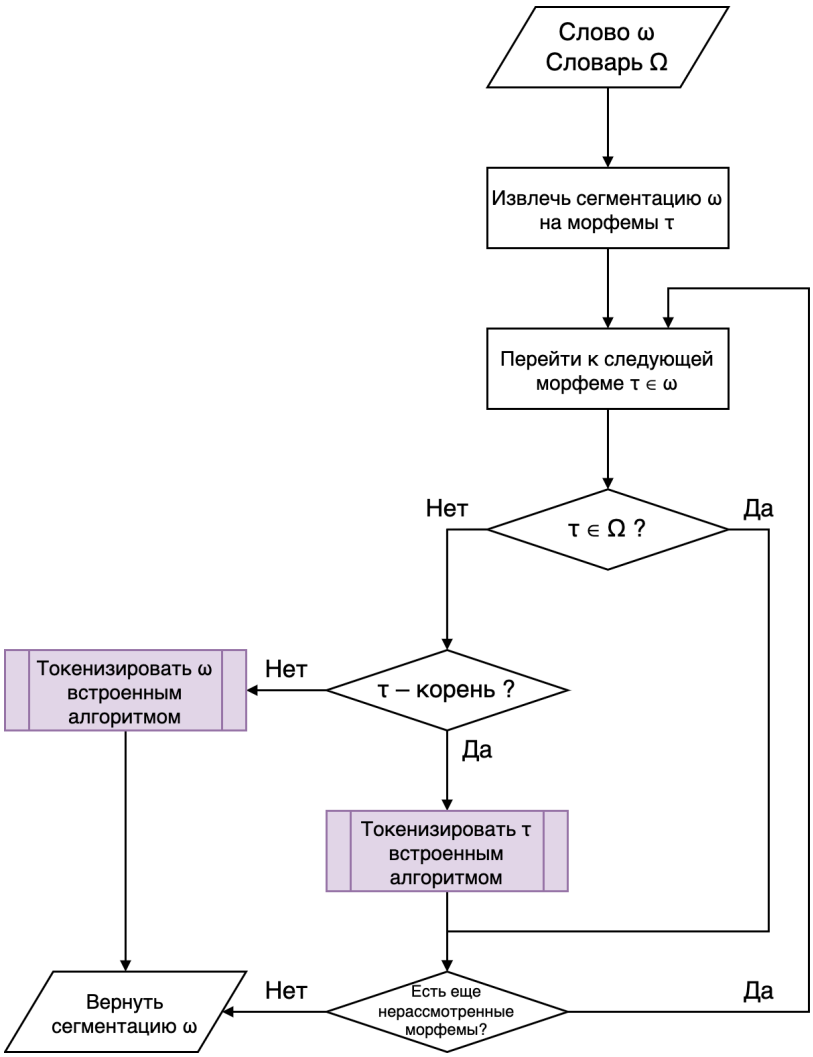



Рисунок 7: Блок-схема разработанной процедуры токенизации слов

Примеры токенизации слов из датасета RuRareWords с использованием разработанной процедуры токенизации и словарей предобученных языковых модели Sber RuBERT и DeepPavlov RuBERT представлены в таблицах 11 и 12 соответственно (см. столбец 2). Из таблиц видно, что для слов «гиперграни», «изометрия», «биоценоз», «идиоплазма» и «фоторецептор» удалось получить морфологически корректную сегментацию для обоих словарей моделей. Для слова «аддитивность» морфологически корректную сегментацию удалось получить только с использованием словаря модели DeepPavlov RuBERT, а

с использованием словаря модели Sber RuBERT корень «аддитив» был дополнительно токенизирован встроенным алгоритмом токенизации. Корень «аксон» слова «аксонометрия» был дополнительно токенизирован для обоих словарей предобученных языковых моделей. Для слова «диссимиляция» использовалась исходная токенизация встроенным алгоритмом, поскольку в словарях моделей отсутствует суффикс «яци» как отдельный токен. В целом видно, что токенизация с использованием морфологически корректной сегментации сохраняет морфологическую структуру токенизированного слова.

Таблица 11: Примеры токенизации слов из датасета RuRareWords с использованием разработанной процедуры токенизации и словаря модели Sber RuBERT

| | Слово | $s_{\text{Del}\times\text{SberRuBERT}}(x)$ | $s_{\text{SberRuBERT}}(x)$ |
|---|---|---|---|
| Математика | аддитивность | ад, ##дит, ##ив, ##н, ##ост, ##ь | ад, ##дит, ##ив, ##ность |
| | аксонометрия | ак, ##сон, ##о, метр, ##и, ##я | ак, ##соном, ##ет, ##рия |
| | гиперграньь | гипер, грань | гипер, ##гран, ##ь |
| | изометрия | изо, метр, ##и, ##я | изо, ##метрия |
| Биология | биоценоз | био, цен, ##оз | био, ##цено, ##з |
| | диссимиляция | дисси, ##мил, ##я, ##ция | дисси, ##мил, ##я, ##ция |
| | идиоплазма | идио, плазм, ##а | идио, ##пла, ##зма |
| | фоторецептор | фото, рецепт, ##ор | фоторе, ##цеп, ##тор |

Таблица 12: Примеры токенизации слов из датасета RuRareWords с использованием разработанной процедуры токенизации и словаря модели DeepPavlov RuBERT

| | Слово | $s_{\text{Del}\times\text{DeepPavlovRuBERT}}(x)$ | $s_{\text{DeepPavlovRuBERT}}(x)$ |
|---|---|---|---|
| Математика | аддитивность | аддитив, ##н, ##ост, ##ь | аддитив, ##ность |
| | аксонометрия | акс, ##он, ##о, метр, ##и, ##я | акс, ##оном, ##етрия |
| | гиперграньь | гипер, грань | гиперг, ##рань |
| | изометрия | изо, метр, ##и, ##я | изом, ##етрия |
| Биология | биоценоз | био, цен, ##оз | био, ##цен, ##оз |
| | диссимиляция | дисс, ##ими, ##ляция | дисс, ##ими, ##ляция |
| | идиоплазма | идио, плазм, ##а | иди, ##оплаз, ##ма |
| | фоторецептор | фото, рецепт, ##ор | фотор, ##ецеп, ##тор |

## 4.4 Эксперименты с моделью

Для проведения экспериментов был написан программный модуль на языке программирования Python 3 с использованием фреймворка машинного обучения PyTorch [29]. Основные компоненты модуля:

1. Dataset – конвертация данных из датасета RuRareWords во внутренний формат PyTorch и токенизация слов на токены разработанной процедурой.

2. Collator – унификация длин входной последовательности токенов в батче (пакете/партии данных), т.е. дополнение последовательности `[PAD]` токенами. Размер батча – гиперпараметр, настраиваемый при обучении.

3. BertModel – описание архитектуры модели DelRuBERT, включающей в себя архитектуру BERT и полносвязного нейросетевого классификатора, с использованием реализации слоев из фреймворка PyTorch.

4. Trainer – основной цикл итеративного обучения модели, включающий в себя вычисление функции ошибок и метрик качества на тренировочном и валидационном наборах данных на каждой итерации.

Данные из датасета RuRareWords разбивались на тренировочные, валидационные и тестовые в соотношении 80% : 10% : 10%. Для оценки качества классификации использовалось несколько метрик. Метрики для задачи классификации формулируются в терминах ошибок (несоответствий) классификации. Матрица ошибок (confusion matrix) бинарной классификации выглядит следующим образом:

| | | Реальный класс | |
|---|---|---|---|
| | | Positive (1) | Negative (0) |
| Предсказанный класс | Positive (1) | True Positive (TP) | False Positive (FP) |
| | Negative (0) | False Negative (FN) | True Negative (TN) |

На основе данной матрицы получаются следующие метрики:

- $\text{precision} = \dfrac{TP}{TP + FP}$ – точность – доля объектов с верно предсказанным положительным классом среди всех объектов с предсказанным положительным классом.
- $\text{recall} = \dfrac{TP}{TP + FN}$ – полнота (чувствительность) – доля объектов с верно предсказанным положительным классом среди всех объектов положительного класса.
- $\text{F}_1 = 2 \cdot \dfrac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \dfrac{TP}{TP + \frac{1}{2}(FP + FN)}$ – среднее гармоническое точности и полноты.

В силу сбалансированности количества примеров для каждого тематического класса, в качестве метрики для оценки качества классификации были выбраны показатели ROC AUC, равный доле верно упорядоченных пар объектов, и $F_1$-мера.

Для оценки качества многоклассовой классификации рассматривалась:

- weighted macro $F_1$-мера, равная взвешенной комбинации $F_1$-меры для каждого тематического класса с долей примеров класса в качестве весов.
- weighted ROC AUC, равный взвешенной комбинации значений ROC AUC для каждого тематического класса с долей примеров класса в качестве весов в подходе «Один против всех», когда рассматриваются объекты одного зафиксированного класса против объектов всех остальных классов.

В таблице 13 представлены результаты экспериментов с оценкой качества бинарной классификации на тестовом наборе данных для тематических классов «математика» и «биология», проведенных с целью выбрать нейросетевую архитектуру для классификатора. Лучшее качество для всех рассматриваемых предобученных языковых моделей независимо от способа токенизации показала двухслойная нейронная сеть с функцией активации GeLU и слоем Dropout между полносвязными слоями. В дальнейшем результаты экспериментов будут приведены для данной архитектуры классификатора.

Таблица 13: Оценка качества разных конфигураций классификатора

| Языковая модель | Классификатор | $AUC_{test}$ | |
|---|---|---|---|
| | | встроенная токенизация | токенизация DelRuBERT |
| Sber RuBERT | 1-слойный | 0.785 | 0.811 |
| | 2-слойный с ReLU | 0.793 | 0.829 |
| | 2-слойный с LeakyReLU | 0.792 | 0.833 |
| | **2-слойный с GeLU** | **0.798** | **0.837** |
| DeepPavlov RuBERT | 1-слойный | 0.786 | 0.797 |
| | 2-слойный с ReLU | 0.783 | 0.801 |
| | 2-слойный с LeakyReLU | 0.788 | 0.807 |
| | **2-слойный с GeLU** | **0.791** | **0.809** |

В случае решения задачи бинарной классификации, слова, которые одновременно попадали в обе категории, не рассматривались в обучении. Например: инверсия, параллелизм, транспозиция для тематик «математика» и «биология». Отметим, что таких

слов было очень мало ($< 0.5\%$ от всех слов тематического класса). В случае решения задачи многоклассовой классификации никакие изменения датасета не производились.

В таблице 14 представлены результаты экспериментов с оценкой качества бинарной и многоклассовой (4 класса) классификации на тестовом наборе данных. Как видно из таблицы 14 (3 и 4 столбцы), во всех задачах и для всех рассматриваемых предобученных языковых моделей, модель DelRuBERT с использованием токенизации слов русского языка на базе встроенного словаря маскированной языковой модели с учетом морфологически корректной сегментации значимо превосходит модели со встроенными алгоритмами токенизации.

Таблица 14: Оценка качества разных конфигураций модели DelRuBERT

| Языковая модель | Алгоритм токенизации | Тип классификации | ROC AUC | $F_1$-мера |
|---|---|---|---|---|
| Sber RuBERT | BPE | Математика Vs Биология | 0.798 | 0.827 |
| | | Юриспруденция Vs Экономика | 0.876 | 0.872 |
| | | Многоклассовая | 0.753 | 0.801 |
| | **Токенизация DelRuBERT** | Математика Vs Биология | **0.837** | **0.851** |
| | | Юриспруденция Vs Экономика | **0.901** | **0.923** |
| | | Многоклассовая | **0.780** | **0.822** |
| DeepPavlov RuBERT | WordPiece | Математика Vs Биология | 0.791 | 0.812 |
| | | Юриспруденция Vs Экономика | 0.896 | 0.887 |
| | | Многоклассовая | 0.774 | 0.797 |
| | **Токенизация DelRuBERT** | Математика Vs Биология | **0.809** | **0.843** |
| | | Юриспруденция Vs Экономика | **0.912** | **0.909** |
| | | Многоклассовая | **0.795** | **0.828** |

Код для построения модели DelRuBERT и проведения экспериментов написан на языке программирования Python 3 с использованием фреймворка машинного обучения PyTorch и доступен по ссылке в git-репозитории: Berezniker/DelRuBERT.

# 5 Заключение

В настоящей работе:

1. Изучены архитектуры современных нейросетевых маскированных языковых моделей и применяемые в них алгоритмы токенизации.

2. Построен датасет RuRareWords, состоящий из редких и морфологически сложных слов, встречающихся в русскоязычных текстах, с разбором этих слов на морфемы, а также токенизацией встроенными алгоритмами (BPE и WordPiece) рассматриваемых языковых моделей для областей: «математика», «биология», «юриспруденция», «экономика».

3. Разработана процедура токенизации слов на базе встроенного словаря предобученной маскированной языковой модели с учетом морфологически корректной сегментации. Построена и обучена модель-классификатор DelRuBERT с использованием разработанной процедуры токенизации на базе предобученной маскированной языковой модели для русского языка.

4. Проведены эксперименты по оценке влияния способа токенизации на качество тематической классификации слов русского языка маскированными языковыми моделями, которые продемонстрировали значимое улучшение качества тематической классификации слов по метрике ROC AUC и $F_1$ при использовании токенизации с учетом морфологически корректной сегментации слов.

Продемонстрированное значимое улучшение качества тематической классификации редких и морфологически сложных слов русского языка означает, что токенизация с учетом морфологически корректной сегментации слов может быть использована для улучшения качества работы систем классификации на базе предобученной маскированной языковой модели BERT.

По результатам проделанной работы был сделан доклад с публикацией тезисов на международном молодежном научном форуме «Ломоносов-2023» [32].

# Литература


[1] Natural language processing [Электронный ресурс]. – Электрон.дан. – URL: https://en.wikipedia.org/wiki/Natural_language_processing.

[2] Lena Voita Language Modeling [Электронный ресурс]. – Электрон.дан. – URL: https://lena-voita.github.io/nlp_course/language_modeling.html.

[3] Chain rule (probability) [Электронный ресурс]. – Электрон.дан. – URL: https://en.wikipedia.org/wiki/Chain_rule_(probability).

[4] Mikolov T. et al. Efficient estimation of word representations in vector space //arXiv preprint arXiv:1301.3781. – 2013.

[5] 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.

[6] Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.

[7] Sennrich R., Haddow B., Birch A. Neural machine translation of rare words with subword units //arXiv preprint arXiv:1508.07909. – 2015.

[8] Schuster M., Nakajima K. Japanese and korean voice search //2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). – IEEE, 2012. – С. 5149-5152.

[9] Hofmann V., Pierrehumbert J. B., Schütze H. Superbizarre Is Not Superb: Derivational Morphology Improves BERT's Interpretation of Complex Words //arXiv preprint arXiv:2101.00403. – 2021.

[10] Sutskever I., Vinyals O., Le Q. V. Sequence to sequence learning with neural networks //Advances in neural information processing systems. – 2014. – Т. 27.

[11] Hochreiter S., Schmidhuber J. Long short-term memory //Neural computation. – 1997. – Т. 9. – №. 8. – С. 1735-1780.

[12] Cho K. et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation //arXiv preprint arXiv:1406.1078. – 2014.

[13] Bahdanau D., Cho K., Bengio Y. Neural machine translation by jointly learning to align and translate //arXiv preprint arXiv:1409.0473. – 2014.

[14] Luong M. T., Pham H., Manning C. D. Effective approaches to attention-based neural machine translation //arXiv preprint arXiv:1508.04025. – 2015.

[15] Vaswani A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – Т. 30.

[16] Gage P. A new algorithm for data compression //C Users Journal. – 1994. – Т. 12. – №. 2. – С. 23-38.

[17] Radford A. et al. Language models are unsupervised multitask learners //OpenAI blog. – 2019. – Т. 1. – №. 8. – С. 9.

[18] Liu Y. et al. Roberta: A robustly optimized bert pretraining approach //arXiv preprint arXiv:1907.11692. – 2019.

[19] Kudo T. Subword regularization: Improving neural network translation models with multiple subword candidates //arXiv preprint arXiv:1804.10959. – 2018.

[20] Yang Z. et al. Xlnet: Generalized autoregressive pretraining for language understanding //Advances in neural information processing systems. – 2019. – Т. 32.

[21] Lan Z. et al. Albert: A lite bert for self-supervised learning of language representations //arXiv preprint arXiv:1909.11942. – 2019.

[22] Bilmes J. A. et al. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models //International computer science institute. – 1998. – Т. 4. – №. 510. – С. 126.

[23] Provilkov I., Emelianenko D., Voita E. Bpe-dropout: Simple and effective subword regularization //arXiv preprint arXiv:1910.13267. – 2019.

[24] Hofmann V., Pierrehumbert J. B., Schütze H. DagoBERT: Generating derivational morphology with a pretrained language model //arXiv preprint arXiv:2005.00672. – 2020.

[25] Kuratov Y., Arkhipov M. Adaptation of deep bidirectional multilingual transformers for russian language //arXiv preprint arXiv:1905.07213. – 2019.

[26] sberbank-ai/ruBert-base · Hugging Face [Электронный ресурс]. – Электрон.дан. – URL: https://huggingface.co/sberbank-ai/ruBert-base.

[27] 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. – С. 78-84.

[28] Srivastava N. et al. Dropout: a simple way to prevent neural networks from overfitting //The journal of machine learning research. – 2014. – Т. 15. – №. 1. – С. 1929-1958.

[29] PyTorch [Электронный ресурс]. – Электрон.дан. – URL: https://pytorch.org/.

[30] CrossMorphy [Электронный ресурс]. – Электрон.дан. – URL: https://github.com/alesapin/XMorphy

[31] Bolshakova E., Sapin A. Building dataset and morpheme segmentation model for Russian word forms //Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference “Dialogue. – 2021. – С. 154-161.

[32] Березникер А.В. Морфологическая сегментация для предобученных маскированных языковых моделей // Международный молодежный научный форум «Ломоносов-2023»: Тезисы докладов.

# Приложение А

# Примеры слов из датасета RuRareWords

## Математика


```
{
    "lemma": "аксонометрия",
    "xmorphy": {
        "phem_info": "аксон:ROOT/о:LINK/метр:ROOT/и:SUFF/я:END",
        "tokens": ["аксон", "##о", "##метр", "##и", "##я"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["ак", "##соном", "##ет", "##рия"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["акс", "##оном", "##етрия"]
    }
},
{
    "lemma": "локсодрома",
    "xmorphy": {
        "phem_info": "локс:ROOT/о:LINK/дром:ROOT/а:END",
        "tokens": ["локс", "##о", "##дром", "##а", "##ий"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["лок", "##со", "##дрома"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["лок", "##со", "##д", "##рома"]
    }
{
    "lemma": "эквидистантность",
    "xmorphy": {
        "phem_info": "эквидистант:ROOT/н:SUFF/ост:SUFF/ь:END",
        "tokens": ["эквидистант", "##н", "##ост", "##ь"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["экви", "##дист", "##ант", "##ность"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["экв", "##идис", "##тан", "##тно", "##сть"]
    }
}
```

## Биология


```
{
    "lemma": "абиогенез",
    "xmorphy": {
        "phem_info": "а:PREF/био:ROOT/ген:ROOT/ез:SUFF",
        "tokens": ["а", "##био", "##ген", "##ез"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["аби", "##о", "##генез"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["аб", "##ио", "##ген", "##ез"]
    }
},
{
    "lemma": "рекапитуляция",
    "xmorphy": {
        "phem_info": "ре:PREF/капитул:ROOT/яци:SUFF/я:END",
        "tokens": ["ре",  "##капитул", "##яци", "##я"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["река", "##пит", "##ул", "##я", "##ция"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["река", "##пит", "##уляция"]
    }
},
{
    "lemma": "фоторецептор",
    "xmorphy": {
        "phem_info": "фото:ROOT/рецепт:ROOT/ор:SUFF/ы:END",
        "tokens": ["фото", "##рецепт", "##ор", "##ы"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["фоторе", "##цеп", "##торы"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["фотор", "##ецеп", "##торы"]
    }
}
```

## Юриспруденция


```
{
    "lemma": "аболиционизм",
    "xmorphy": {
        "phem_info": "аболици:ROOT/он:SUFF/изм:SUFF",
        "tokens": ["аболици", "##он", "##изм"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["або", "##лиц", "##и", "##они", "##зм"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["аб", "##оли", "##ционизм"]
    }
},
{
    "lemma": "менажировать",
    "xmorphy": {
        "phem_info": "менаж:ROOT/ир:SUFF/ова:SUFF/ть:END",
        "tokens": ["менаж", "##ир", "##ова", "##ть"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["мен", "##а", "##жир", "##ова", "##ть"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["мен", "##ажи", "##ровать"]
    }
},
{
    "lemma": "стигматизация",
    "xmorphy": {
        "phem_info": "стигм:ROOT/ат:SUFF/из:SUFF/аци:SUFF/я:END",
        "tokens": ["стигм", "##ат", "##из", "##аци", "##я"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["сти", "##гма", "##тизация"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["стиг", "##мати", "##зац", "##ия"]
    }
}
```

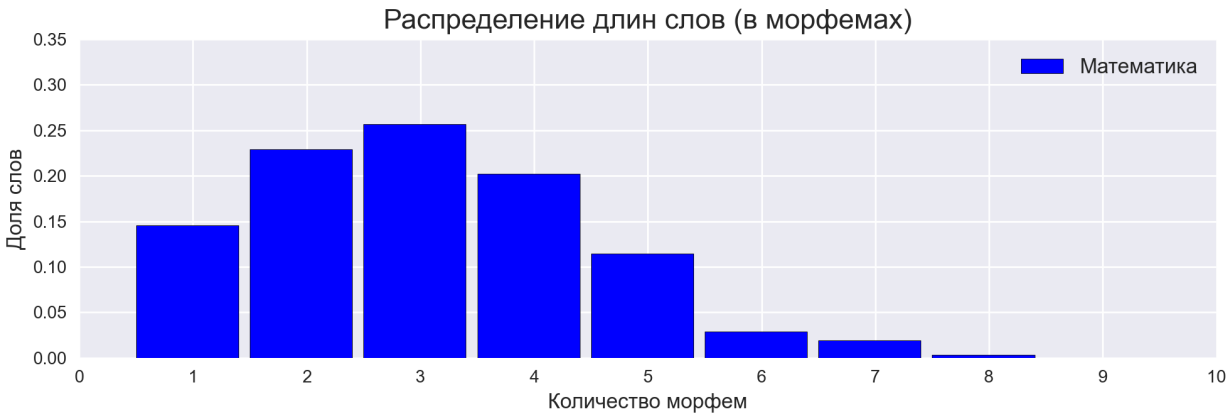
## Экономика

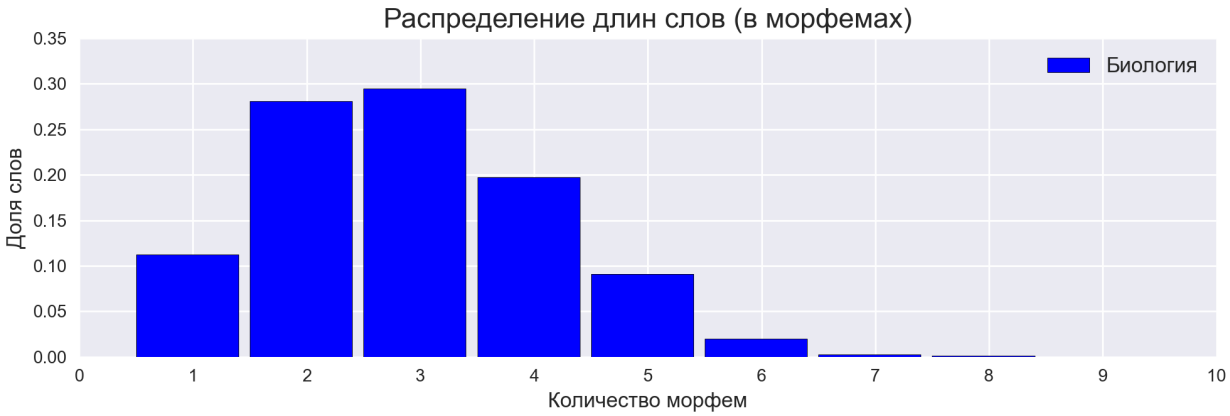

```
{
    "lemma": "валоризация",
    "xmorphy": {
        "phem_info": "валор:ROOT/из:SUFF/аци:SUFF/я:END",
        "tokens": ["валор", "##из", "##аци", "##я"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["вал", "##ори", "##зация"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["вал", "##ори", "##зац", "##ия"]
    }
},
{
    "lemma": "ревальвация",
    "xmorphy": {
        "phem_info": "ре:PREF/вальв:ROOT/аци:SUFF/я:END",
        "tokens": ["ре", "##вальв", "##аци", "##я"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["рев", "##аль", "##вация"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["рев", "##альва", "##ция"]
    }
},
    "lemma": "таргетирование",
    "xmorphy": {
        "phem_info": "таргет:ROOT/ир:SUFF/ова:SUFF/ни:SUFF/е:END",
        "tokens": ["таргет", "##ир", "##ова", "##ни", "##е"]
    },
    "sberbank-ai/ruBert-base": {
        "tokens": ["таргети", "##рование"]
    },
    "DeepPavlov/rubert-base-cased": {
        "tokens": ["тар", "##гет", "##ирование"]
    }
}
```

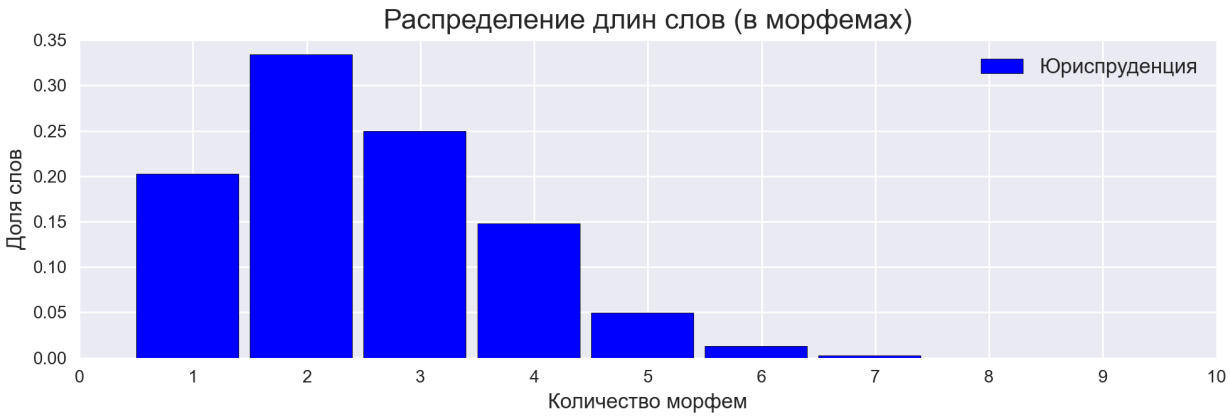
# Приложение Б

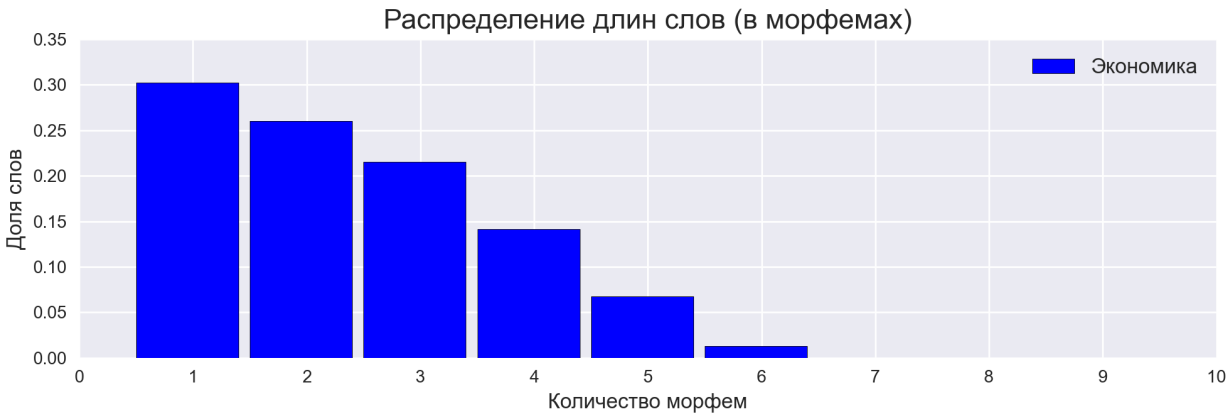
# Распределение длин слов из датасета RuRareWords





Распределение длин слов (в морфемах)
Математика
Доля слов
Количество морфем


Распределение длин слов (в морфемах)
Биология
Доля слов
Количество морфем


Распределение длин слов (в морфемах)
Юриспруденция
Доля слов
Количество морфем


Распределение длин слов (в морфемах)
Экономика
Доля слов
Количество морфем