Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Московский физико-технический институт
(государственный университет)»

Факультет управления и прикладной математики

Кафедра информатики и вычислительной математики

# Детекция и классификация грамматических ошибок в письменной речи

Выпускная квалификационная работа
(магистерская диссертация)

Направление подготовки: 03.03.01 Прикладные математика и физика

Выполнил:
студент 273 группы _____ Емельянов Ярослав Игоревич

Научный руководитель:
д. ф.-м. н. профессор _____ Рязанов Владимир Васильевич

Москва
2018

# Содержание


**1 Введение** **4**

1.1 Постановка задачи и историческая справка . . . 4

1.1.1 Постановка задачи . . . 4

1.1.2 Влияние лингвистической неоднозначности . . . 4

1.1.3 Оценка качества модели . . . 6

1.1.4 Используемые данные . . . 8

1.2 Подходы к решению . . . 11

1.2.1 Решения, основанные на правилах . . . 11

1.2.2 Статистические языковые модели . . . 12

1.2.3 Нейронные языковые модели . . . 13

1.2.4 Рекуррентный слой . . . 16

**2 Нейронный машинный перевод** **19**

2.1 Статистический машинный перевод . . . 19

2.2 Кодировщик-декодировщик . . . 20

2.3 Внимание . . . 21

2.3.1 Функции потерь . . . 22

2.3.2 Отбор гипотез . . . 23

**3 Эксперимент** **25**

3.1 Предварительное исследование: синтетический язык . . . 25

3.1.1 Регулярные языки . . . 25

3.2 Предварительное исследование: английский язык . . . 25

3.2.1 Статистическая языковая модель . . . 25

3.2.2 Нейронная языковая модель . . . 28

3.3 Нейронный машинный перевод . . . 31

3.3.1 Количественная и качественная оценка . . . 31

3.4 Перенос знаний . . . 33

3.4.1 Текущее применение в обработке естественного языка . . . 33

3.4.2 Результаты . . . 34

3.4.3 Качественная оценка и обсуждение . . . 34

3.5 Решение задачи классификации . . . 36

**4 Заключение** **38**

4.1 Резюме . . . 38

4.2 Полученные результаты . . . 38

4.3 Дальнейшая работа . . . 38

# Аннотация

При передаче любой информации неизбежны ошибки.

Возникают они и при передаче информации, выраженной на естественном языке. В современном мире количество ошибок, допускаемых в устной и особенно письменной речи, только растет [2], что негативно сказывается на процессе приема и передачи информации. Помимо этого, грамматические ошибки в текстах, таких как резюме или заявление о приеме на работу, отрицательно коррелируют с вероятностью быть трудоустроенным, поскольку грамотность является важным требованием на многих должностях [5]. Дополнительным фактором, усугубляющим положение дел, служит глобализация, благодаря которой большое количество людей получили возможность коммуницировать на не своих родных языках. Существующие системы машинного перевода (например, [59] или [9]) недостаточно правильно переводят письменную речь и испытывают проблемы с устной речью.

Этим обоснована необходимость как повышения грамотности носителей языка (L1) и изучающих язык (L2), так и исправления ошибок и опечаток в письменной речи, с тем чтобы сделать ее более легко усваиваемой, особенно изучающими язык.

Решение этих задач состоит в организации образовательного процесса изучающих язык, особенно взрослых иностранцев, а также создании инструмента автоматической коррекции ошибок в письменной речи.

Автоматическая коррекция ошибок (spellchecking) – задача, состоящая в том, чтобы сделать заданный фрагмент устной или письменной речи грамматически и семантически корректным, минимально изменив его смысл.

Распространенные на текущий момент технологии автоматической коррекции [42] основаны на правилах и не могут использоваться безотносительно языка, серьезно увеличивая требования к ресурсам на создание, улучшение и поддержку таких систем. В свою очередь, алгоритмы на основе машинного обучения в основной своей массе требовательны к данным, а разметка обучающих выборок для таких моделей является трудоемким процессом.

В данной работе рассматриваются вопросы разработки моделей коррекции языка на основе обучения с учителем и без учителя, а также использования моделей обучения с учителем в ситуациях с сильно ограниченными данными.

В предлагаемой работе поставлены следующие цели:

1. Оценить потенциал обучения без учителя для задачи поиска ошибок.
2. Исследовать потенциал систем машинного перевода в задаче по коррекции ошибок.
3. Исследовать возможность переноса знаний на языки с малым количеством размеченных данных.

На сегодняшний день существует несколько коммерческих и некоммерческих решений для проверки орфографии и грамматики. Однако, в большинстве случаев эти решения работают только для одного языка, чаще всего английского, и имеют ограниченные возможности по анализу ошибок как в грамматике, так и в семантике. Помимо этого, вопрос о том, как ошибочная речь влияет на языковые модели, не изучен до конца.

В предлагаемой работе предложены и реализованы следующие идеи:

1. На примере искусственных языков показывается, как количество и качество ошибок влияет на "память" нейронной сети – ее скрытое состояние – и делается вывод об ограниченности подхода обучения без учителя, который подтверждается масштабными экспериментами на корпусе FCE.

2. Показывается, что алгоритмы нейронного машинного перевода предоставляют возможность для переноса знаний, таким образом, появляется возможность разрабатывать алгоритмы для языков с экстремально малым количеством размеченных данных, на примере переноса модели с немецкого на английский.

3. Приводится пример алгоритма классификации ошибок, принимающий на вход исходный и исправленный тексты, работающий на основе парсера, и проводится оценка точности такого алгоритма.

Ошибки, допускаемые в естественном языке, делятся на несколько типов [12], мы будем пользоваться типизацией, принятой в корпусе NUCLE, о котором пойдет речь ниже.

Дополнительный интерес представляет вопрос, может ли полученный алгоритм быть использован в реальных задачах как для обучения людей, так и для дообучения существующих алгоритмов. Предложенные в работе методы планируется применить в продукте для изучения английского и испанского языков.

# Введение

## 1.1 Постановка задачи и историческая справка

### 1.1.1 Постановка задачи

Дан текст на языке $L^*$: $S = s_1, \ldots, s_n, < EOS >$, представленный в виде последовательности токенов.

Текст может содержать грамматические ошибки – это ошибки в структуре языковой единицы: слова, словосочетания или предложения, т. е. нарушение какой-либо грамматической нормы – словообразовательной, морфологической, синтаксической.
К грамматическим ошибкам относится достаточно широкий класс всех совершаемых ошибок, они встречаются наиболее часто в различных языках.

Два широких класса ошибок – лексические и семантические – не относятся к грамматическим. Поиск и устранение таких ошибок является другим важным вопросом, который не затрагивается в данной работе.

Задача **детекции** ошибок – нахождение по $S$ маски $b = b_1, \ldots, b_n$ , где

$$b_i = \begin{cases} 1, & s_i, s_{i+1} \text{ нет грамматической ошибки} \\ 0, & \text{иначе} \end{cases}$$

Такое определение необходимо для того, чтобы можно было описать ошибку, связанную с пропущенным слово: в предложении "I have cat." ошибка будет найдена на 2 позиции, так как между "have" и "cat" должен стоять артикль "the".

Задача **классификации** ошибок – это сопоставление каждому токену, для которого $b_i = 1$, исправления в виде одного или нескольких токенов $C = c_i^1, c_i^2, \ldots, c_i^{n_i}$.

Таким образом, после выполнение обеих процедур наш текст будет иметь вид
$S' = s_1, c_1^1, c_1^2, \ldots, c_1^{n_i}, s_2, \ldots, s_n, < EOS >$

Заметим, что под токенами в постановке задачи могут пониматься как слова, так и буквы или буквенные n-граммы (то есть последовательность из $n$ символов, идущих подряд). Дополнительный EndOfSentence токен (<EOS> в дальнейшем) нужен для того, чтобы постановка задачи всегда была корректной ($s_{i+1}$ всегда существовал). В данной работе рассматриваются как модели, считающие токеном отдельное слово, так и модели, работающие с более мелкими единицами. Преимущество моделей, работающих на уровне слов (word-level моделей) в том, что слова (в отличие от буквенных $n$-грамм) всегда содержат какой-то смысл, и поэтому интерпретируемость полученных моделей будет выше.

### 1.1.2 Влияние лингвистической неоднозначности

Как и во многих других задачах обработки языка, контекст (встреченная ранее в тексте информация, позволяющая устранить лингвистическую неоднозначность) серьезно усложняет задачу. Рассмотрим

следующие примеры:

I going to school.

Данное предложение может быть исправлено как "I am going to school." или "I go to school.". Каждое из исправлений имеет одинаковое корректорское расстояние на уровне слов, однако их смысл *сильно* различается: одно имеет отношение к настоящему моменту, и может быть употреблено в контексте "Where are you?", в то время как другое описывает постоянно повторяющийся процесс и может быть употреблено в контексте "Do you study at home or at school?". Легко видеть, что контексты для двух вышеописанных исправлений не только не являются взаимозаменяемыми, но также противоречат друг другу в большой степени. Данный пример служит иллюстрацией к проблеме многозначности предложения в зависимости от контекста, которая может быть усугублена разнообразными лексическими средствами (такими как эллипсис или кореферентность (coreference)).

Другим важным примером многозначного контекста служат предложения с непроективными деревьями разбора, то есть с деревьями, в которых есть пересекающиеся вершины либо есть завимисости, проходящие над корнем приложения. Синтаксический разбор выполнен с помощью библиотеки SpaCy [23].

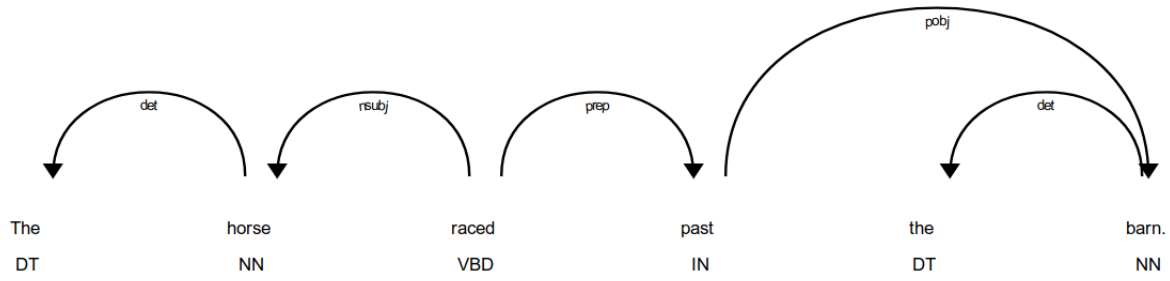



Рисунок 1: Дерево разбора предложения "The horse raced past the barn."

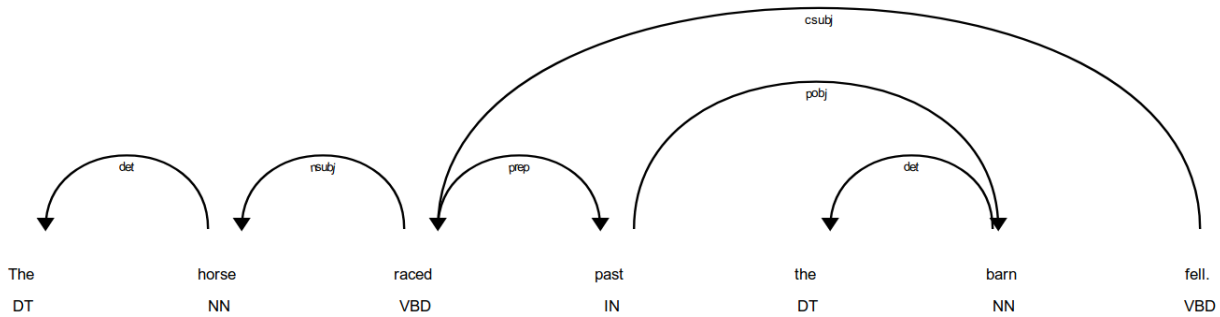



Рисунок 2: Дерево разбора предложения "The horse raced past the barn fell."

Представим себе, что данное предложение (или какое-либо похожее на него по синтаксической структуре) написал человек с не очень высоким уровнем английского. Тогда с большой вероятностью оно не только лишь не будет вполне грамматичным, а также не будет вполне семантически корректным. Это дополнительно увеличивает число возможных вариантов исправления, выбор из которых опять же невозможен без контекста.

С целью упрощения задачи как на уровне модели, так и на уровне сбора и предобработки данных, в данной работе контекст за пределами рассматриваемого предложения не принимается в рассмотрение. Однако, это не вступает в противоречие с оценочными метриками, поскольку они все вычисляются для каждого предложения по отдельности и игнорируют какие-либо взаимосвязи между предложениями.

### 1.1.3 Оценка качества модели

Рассмотрим предложение

I going to kitchen.

Попросим нескольких аннотаторов независимо исправить его:

1. I **go** to the kitchen.
2. I **am** going to **the** kitchen.

Пусть наша система выдала ответ "I am going to kitchen." Рассмотрим, насколько хорошо будет оценено такое исправление несколькими различными метриками:

#### Корректорское расстояние

Исправления происходят на уровне слов, то есть $T$ отличается от $S$ некоторым набором токенов. Например, для предложения There is no a person who have never seen ice. Следующие исправления будут правильными: $g = \{\text{a person} \rightarrow \text{person}, \text{have} \rightarrow \text{has}\}$

Пусть наша система произвела список исправлений $e$ в таком же формате: Precision и Recall (наряду с ложноположительными и ложноотрицательными срабатываниями) полностью описывают confusion matrix.

$$P = \frac{\sum_{i=1}^{n} |g_i \cap e_i|}{\sum_{i=1}^{n} |g_i|}$$

$$R = \frac{\sum_{i=1}^{n} |g_i \cap e_i|}{\sum_{i=1}^{n} |e_i|}$$

На практике для удобства часто используют функции от этих метрик, такие как $F_1$ score: $[F_1 = 2 \cdot \frac{P \cdot R}{P+R}]$

#### BLEU

Метрика BLEU [47] является одной из наиболее популярных метрик для машинного перевода. Для ее вычисления необходимо посчитать все n-граммы, $1 \leq n \leq n_{max}$, и затем усреднить вхождения всех $n$-грамм из референсного ответа в тестовый:

$$s = \frac{\sum_{c \in \{candidates\}} \sum_{n-gram \in c} N'(n-gram)}{\sum_{c' \in \{candidates\}} \sum_{n-gram' \in c'} N(n-gram')},$$

где $N(x)$ – число вхождений n-граммы в предложение, а

$$N'(x) = min(N(x), N_{max},$$

$N_{max}$) – наперед заданная константа.

### GLEU

Было замечено, что результаты работы модели улучшаются по мнению людей, если в метрике учитывать не только перекрывающиеся в исправлении и Gold Label исправлении n-граммы, а также с отрицательным весом учитывать те n-граммы, которые были предложены моделью, но не встретились в референсном сравнении.

Более того, в ходе проведеннного в [7] исследования, целью которого было исследовать корреляции между различными этими метриками и оценкой асессоров, обнаружилось, что существующие метрики плохо скоррелированы с мнением людей.

В этих условиях была показана необходимость более точной метрики. Такая метрика получила название GLEU [41] и стала одной из наиболее прогрессивных метрик для автоматической оценки качества машинного перевода. В [45] предложена модификация, позволяющая использовать метрику в случае нескольких асессоров. В [54] эта метрика использована напрямую для достижения более высоких результатов на CONLL 2014.

Версия GLEU+ дополнительно учитывает наличие нескольких правильных переводов, а не просто усреднение:

$$p_n^* = \frac{\left( \sum\limits_{ngram \in \{C \cap R\}} count_{C,R}(ngram) - \sum\limits_{ngram \in \{C \cap S\}} \max\left[0, count_{C,S}(ngram) - count_{C,R}(ngram)\right] \right)}{\sum\limits_{ngram \in \{C\}} count(ngram)}$$

### Необъективность метрик

Использование автоматических метрик сильно уменьшает ресурсы, требуемые как на обучение моделей, так и на их сравнение друг с другом, однако имеет ряд недостатков:

1. Как было сказано выше, $n$-граммные метрики вознаграждают модели за принятие *локально* правильных решений, что может приводить к плохому глобальному результату. Например, предложение "The horse raced past the barn fallen." может быть исправлено как "The horse raced past the barn **have** fallen.", что является локально хорошим решением, однако не является правильным решением, так как в получившемся предложении нарушено согласование времен.

2. В отдельных работах предпринимались попытки обучения метрики (metric learning), показавшие определенный результат, однако сохранение этого результата на больших моделях и выборках не проверено.

### Оценка качества с помощью людей

Традиционно, именно люди выполняют работу по оценке письменной речи. Несмотря на трудоемкость данного метода (скорость обработки одного предложения человеком на несколько порядков ниже, чем компьютером), люди способы не только исправлять ошибки, а еще давать интерпретируемые объяснения ошибок ("здесь нужен артикль the, потому что его всегда нужно ставить перед прилагательным в превосходной степени") и рекомендации ("в сегодняшнем домашнем задании будет 15 предложений на употребление артиклей").

По этой причине использование ручного труда, хотя и дает наиболее высокие результаты, зачастую ограничено очными или дистанционными занятиями и не может быть отмасштабировано на повседневную жизнь, когда человек постоянно общается в письменной или устной форме. Однако размечаемые людьми корпуса являются полезными при разработке автоматических систем, которые могли бы обучаться на

| Источник | This phenomenon opposes the real. |
|---|---|
| Асессор А | This phenomenon opposes (the$\rightarrow \epsilon$ (ArtOrDet)) (real $\rightarrow$ reality (Wform)) |
| Асессор Б | This phenomenon opposes the (real $\rightarrow$ reality (Wform)) |

Таблица 1: Разные исправления предложения "This phenomenon opposes the real".

них.

В [12] авторы обращают внимание на то, что несмотря на стандартизацию типов ошибок и процесса аннотирования (они используют специально разработанный веб-интерфейс для аннотаций), возникают серьезные проблемы с тем, чтобы согласовать даже одинаковые аннотации. Рассмотрим пример:

Для количественной оценки согласованности исправлений разных аннотаторов используется следующая схема:

1. Идентификация – все аннотаторы согласны с наличием ошибки в данных токенах
2. Классификация – все аннотаторы согласны с тегом ошибки
3. Полное совпадение – все аннотаторы согласны с тегом ошибки, а также последовательностью некорректных и корректных токенов.

В примере выше, ассессоры А и Б согласны в 5 из 6 токенов, тем самым согласованность при идентификации достигает 83%.

Для численного выражения согласованности [7] предлагает использовать Cohen's Kappa Coefficient:

$$\kappa = \frac{Pr(a) - Pr(e)}{1 - Pr(e)},$$

$$Pr(a) = \frac{\text{число согласованных токенов}}{\text{Общее число токенов}},$$

$Pr(e)$ – вероятность случайного согласования. В данном случае она велика ввиду несбалансированности классов, поскольку среднее количество ошибок в текстах сравнительно мало.

$$Pr(e) = Pr(A = 1, B = 1) + Pr(A = 0, B = 0) = Pr(A = 1) \cdot Pr(B = 1) + Pr(A = 0) \cdot Pr(B = 0),$$

так как аннотирования производятся независимо. Вероятность аннотирования аннотатором текста, в свою очередь, могут быть выписаны как

$$Pr(A = 1) = \frac{\text{число аннотированных аннотатором А токенов}}{\text{общее число токенов}}$$

Проводя вычисления, эти же авторы сообщают о достаточно низких коэффициентах $\kappa$: от 0.39 до 0.55 в зависимости от типа согласования, при этом наиболее высокий этот коэффициент для классификации, что дополнительно подчеркивает субъективность задачи.

### 1.1.4 Используемые данные

Данные занимают центральное место в машинном обучении, потому как их отсутствие зачастую представляет собой серьезное препятствие для исследований. Долгое время не существовало публичных корпусов и унифицированных метрик для оценки алгоритмов, а собираемые и аннотируемые в национальных исследовательских лабораториях и коммерческих компаниях данные редко становились публично доступными. Ситуация начала меняться лишь сравнительно недавно, когда были объявлены конкурсы на создание лучшей системы коррекции ошибок в рамках CONLL-2013 и CONLL-2014, которые используются для оценки качества моделей и в настоящее время.

### FCE

Данный корпус [61] представляет из себя коллекцию сочинений экзамена ESOL (English as a Second or Other Language, Английский как второй или иной язык), представленных для воспроизведения результатов работы алгоритма автоматической оценки качества письменных текстов. Данная задача проще, чем рассматриваемая в этом дипломе, потому что в качестве выходных данных ожидается всего 1 категориальный признак с небольшим множеством значений, и маленький корпус подходит для получения статистически значимого результата *в ней*, в то же время он оказывается недостаточен для получения статистически значимого результата *в нашей задаче*.

### CONLL-2013

CONLL 2013 Grammatical Error Correction Shared Task [46] стало одним из первых соревнований, в которых был представлен корпус достаточно большого объема. Командам предлагалось реализовать систему коррекции ошибок пяти различных видов:

1. Ошибки в артиклях или определителях
2. Употребление предлогов
3. Форма существительного
4. Форма глагола
5. Согласование подлежащего и сказуемого

Предложения были отобраны из опубликованного ранее NUS Corpus of Learning English (NUCLE) [12], который представляет из себя проаннотированные носителями английского языка сочинения студентов Сингапурского университета.

### CONLL-2014

Новое задание [12] стало развитием предыдущего, и в нем была поставлена задача обрабатывать все ошибки, встречающиеся в NUCLE версии 2.3. Приведем описание возможных типов ошибок:

| Тег | Категория | Пример |
|---|---|---|
| Vt | Время глагола | A university [had conducted → conducted] the survey last year. |
| Vm | Модальный глагол | No one will – would bother to consider a natural balance. |
| V0 | Пропущенный глагол | Ths [may – may be] due to a traditional notion that boys would be the main labor force in a farm family. |
| Vmform | Форма глагола | Will the child blame the parents after he [growing – grows] up? |
| SVA | Согласование сказуемого с подлежащим | The boys [play – plays] soccer. |
| ArtorDet | Артикль или определитель | From the ethical aspect, sex selection technology should not be used in [non-medical – a non-medical] situation. |
| Nn | Форма существительного | sex selection should therefore be used for medical [reason – reasons] and nothing else. |
| Npos | Притяжательное существительное | The education of [mother's – mother] is a significant factor in reducing son preference. |
| Pform | Форма местоимения | 90% of couples seek treatment for family balancing reasons and most of [those – them] want girls. |
| Pref | Ссылающееся местоимение | Moreover, children may find it hard to communicate with [his – their] parents. |
| Wcip | Неправильное словосочетание, идиома или предлог | Singapore, for example, has invested heavily [on – in] establishment of Biopolis. |
| Wa | Акроним | The RFU agreed to handle the World Football Championship to Russia. |
| Wform | Форма слова | Divorce may also result in [addition – additional] stress for the family. |
| Wtone | Тональность | [Isn't – Is it not] what you always dreamed for? |
| Srun | Формирование предложения | |
| Smod | Неправильное управление придаточным | |
| Spar | Параллелизм | [Faced – when we are faced] with the serious risk of crisis, most couples refuse to make children. |
| Sfrag | Фрагментация | The use of sex selection would prevent rather than [contributing – contribute] to a distorted sex ratio. |
| Ssub | Подчиненное предложение | It is the wrong mindset of people that boys are more superior than girls ]should – tha should] be corrected. |

Разметка сделана по принципу "предложение – аннотации". Аннотации дополнительно сгруппированы по аннотировавшим. Пример разметки:

```
S Giant otters is an apex predator .
A 2 3|||SVA|||are|||REQUIRED|||-NONE-|||0
A 3 4|||ArtOrDet|||-NONE-|||REQUIRED|||-NONE-|||0
A 5 6|||NN|||predators|||REQUIRED|||-NONE-|||0
A 1 2|||NN|||otter|||REQUIRED|||-NONE-|||1
```

В данном корпусе предложения размечали один или несколько аннотаторов, информация об этом также содержится. При подсчете официальным инструментом учитывается ближайшее из всех расстояний, что сделано для предотвращения заниженных оценок из-за того, что различные аннотаторы оценивают предложения по-разному.

**Lang-8**

Данный корпус ошибок представляет из себя выгрузку социальной сети Lang-8 [40], в которой добровольцы, являющиеся носителями языка, помогают другим людям изучать тот язык, который знают. Это самый большой открытый корпус английских ошибок, содержащий 237843 пары. Помимо этого, в корпусе представлены ошибки и на других языках:

1. Японский – 185991 пара
2. Китайский (мандаринское наречие)– 28154 пары
3. Корейский – 21779 пар
4. Испанский – 12606 пар
5. Французский – 12392 пары
6. Немецкий – 11111 пара

Стоит заметить, что так как исправления сделаны непрофессионалами (зарегистрироваться в этой социальной сети может каждый, проверки качества знаний не проводится), то этот корпус гораздо более "грязный", чем все остальные. Помимо этого, примерно 58 тысяч предложений не размечены по языкам.

Для наших экспериментов мы используем корпус FCE для предварительных исследований и корпуса NUCLE и Lang-8 для основной части исследований.

## 1.2 Подходы к решению

### 1.2.1 Решения, основанные на правилах

Наиболее ранние подходы появились после работ Ноама Хомского [10], в которых обосновывалось, что синтаксис естественного языка представим в виде контекстно-свободных, а семантика – в виде контекстно-зависимых грамматик. Данные работы получили широкое признание и способствовали увеличению интереса к области машинной обработки естественного языка, и большое количество исследователей начали ставить и решать задачи из этой области. Очень быстро, однако, выявились недостатки формальных онтологий: трудность построения и масштабирования, а также плохая переносимость с языка на язык, что приводило к необходимости трат существенных ресурсов на поддерживание работоспособности алгоритмов в условиях все возрастающего притока данных. Несмотря на это, онтологии и по сей день используются в ряде задач (таких как информационный поиск ), и служат базовым решением для новых задач обработки языка.

**Преимущества**

1. Простота создания в случаях, когда результирующее дерево имеет сравнительно малое количество вершин.

2. Универсальный язык описания [10].

**Недостатки**

1. Большую часть работы необходимо выполнять заново для каждого языка.

2. Трудность поддержки и масштабирования.

3. Низкая робастность и по отношению к неоднозначному контексту и ошибкам, неизбежно возникающим в письменной и устной речи.

### 1.2.2 Статистические языковые модели

Статистические языковые модели появились вследствие невозможности построить удовлетворительные модели языка, основанные на правилах. В их основе лежит подсчет n-грамм для некоторого корпуса, а затем нормировка и использование получившихся в результате подсчета статистик как признака, на основе которого решаются различные задачи: от классификации текстов до генерации речи.

Существенным преимуществом алгоритмов на основе языковых моделей является тот факт, что при рассмотрении узких типов ошибок (например, ошибки употребления предлогов и артиклей в английском языке) удается добиться высокого результата, сильно скоррелированного с мнением асессоров.

**Описание**

Языковые модели описывают язык как условные вероятности $p(w|c)$, где контекст $c$ принимает форму $n$-граммы. Соответственно, для работы модели требуется запомнить все $n$-граммы, встречающиеся в языке. Число таких n-грамм для языка со словарем $V$ составляет $V^n$ и зачастую не может быть вмещено в современные средства памяти. Однако даже без учета этого ограничения существует иная проблема, вызванная разреженностью самих n-грамм в речи.

Воспользуемся законом Ципфа:

$$C_i \sim \frac{1}{i},$$

где $C_i$ – частота символа i после их упорядочивания по частоте

Отсюда видно, что для биграмм закон Ципфа усугубляется:

$$C_{ij} \sim \frac{1}{i \cdot j},$$

Эти факторы в совокупности приводят к тому, что использование языковых моделей для более чем 3- и 4-грамм становится недостижимым ввиду ограничений по памяти и недостатка данных. Для частичного устранения этого эффекта было предложено использовать меньшие $n$-граммы для приближенного расчета вероятностей (back-off) [20], и целевая функция языковой модели записывается как

$$p(w_n|w_1^{n-1}) = p(w_n|w_f^{n-1}) \prod_{i-1}^{f-1} b(w_i^{n-1})$$

, где $b(w_i^{n-1})$ и есть использование меньших $n$-грамм.

**Детекция ошибок с помощью статистической языковой модели.**

Для обучения языковой модели необходимо собрать достаточно большой корпус. После этого можно построить детектор ошибок, используя вероятность $n$-граммы в качестве признака, указывающего на наличие ошибки. В таком режиме модели не требуются данные по исправлениям ошибок, производство которых достаточно трудоемко. Однако, этот подход имеет ряд ограничений:

1. Многие $n$-граммы (например, именованные сущности) могли никогда не встречаться в обучающем корпусе, и наш признак становится неинформативным. Использование выделения именованных сущностей (Named Entity Recongnition) позволяет выделить такие слова, с тем чтобы не учитывать их при построении модели или заменить их на мета-токены, такие как PLACE, LOCATION и т.д. Однако, даже самые современные алгоритмы требовательны к памяти [43], что существенно увеличивает время на обучение всей модели.

2. До 15% ошибок в письменной речи составляют орфографические ошибки (spelling), а поскольку языковую модель необходимо обучать на правильных (или по крайней мере считающихся таковыми) данных, то данные слова не попадают в расчет. Проблему можно было бы решить, перейдя с уровня слов на уровень морфем, однако

3. Многие обучающиеся, особенно на начальных этапах обучения, допускают ошибки, связанные с пропуском служебных или незнакомых им слов, например вспомогательных глаголов, предлогов или артиклей. Таких ошибок достаточно много (до 20%) [15], и они вкупе с неправильно употребленными словами и орфографическими ошибками составляют серьезное препятствие для статистических языковых моделей. Подробный количественный и качественный анализ будет проведен ниже.

4. Наконец, низкая вероятность n-граммы необязательно означает ошибку: в предложении "The old man the boat." 4-грамма "the old man the" крайне маловероятна

5. Высокие требования к ресурсам во время работы: наименее требовательная к памяти 4-граммная модель, обученная на корпусе книг размером в 1.1 гигабайта, имеет вес 7.8 гигабайта.

6. Невозможность детекции ошибок порядка слов: предложение "I am going home is important." имеет сравнительно высокую вероятность по версии статистической языковой модели несмотря на то, что является синтаксически некорректным.

Стоит отметить, что скорость обучения таких моделей – $O(n)$, где $n$ – число слов в корпусе, поэтому основным ограничением их использования выступает оперативная память устройств, где предполагается их размещать во время активной работы.

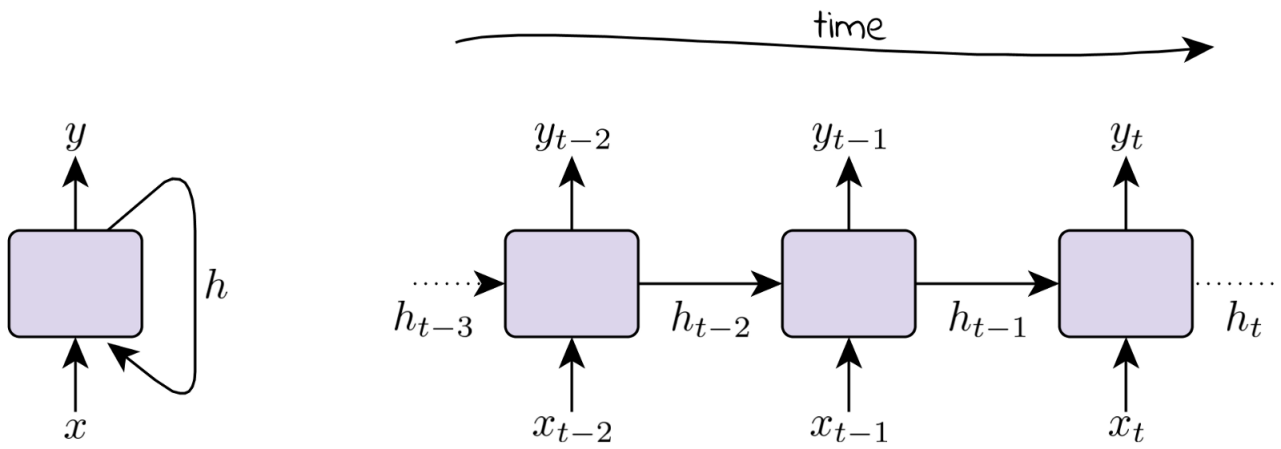
### 1.2.3 Нейронные языковые модели

Нейронные языковые модели получили признание и активное развитие начиная с 2000-х годов, когда работы [4] и [27] превзошли все предыдущие статистические модели, получив наилучшую перплексию для популярного датасета моделирования языка Penn Treebank (Пеннсильванский Сборник Деревьев) [38]. Данные модели обладают следующими основными отличиями от статистических моделей:

1. Использование непрерывных представлений (вложения, Embedding Layer) качестве первого слоя сети для извлечения признаков (семантических либо грамматических) из токенов, которые подаются на вход модели.

2. Использование рекуррентных нейронных сетей (Recurrent Neural Networks), благодаря чему появляется теоретическая возможность учитывать сколько угодно большой контекст: доказано, что рекуррентная сеть может быть использована для моделирования машины Тьюринга.

3. Возможность организовать выходные данные в виде токенов из словаря, быть может, отличного от словаря, используемого для вложений, с помощью специального выходного слоя (Projection Layer). Например, в [**?**] строится иерархия на словаре, что позволяет существенно сократить число параметров в этом слое, ускоряя тренировку модели без существенных потерь в качестве.




Разберем каждое это отличие по отдельности:

1. Вложения
   Вложения (Embeddings) представляют собой матрицу $E \in \mathbf{R}^{v \times i}$, где $i$ – размер вложения. Каждому слову из словаря она сопоставляет некоторый вектор из высокоразмерного пространства $\mathbf{R}^v$. Данная матрица может как обучаться вместе со всей моделью, так и быть предварительно обученной с помощью каких-либо алгоритмов, например, Word2Vec [39].

2. Рекуррентная сеть
   Данные сети представляют из себя функции вида

$$f(x_t, h_t) \to h_{t+1}$$

   В самом простейшем случае

$$f(x_{t+1}, y_{t+1}) = \sigma(Wx_t + Uh_t + b),$$

   где

$$W \in \mathbf{R}^{i \times h},$$

$$U \in \mathbf{R}^{h \times h},$$

$$b \in \mathbf{R}^h$$

   $\sigma$ – нелинейная функция. Обычно используют два вида таких нелинейностей: сигмоида $\sigma(x) = \frac{1}{1+\exp -x}$ и гиперболический тангенс $\tanh(x)$. Можно также использовать непрерывную кусочно-линейную функцию активации, такую как ReLU (Rectified Lilnear Unit) [44]: $ReLu(x) = max(x, 0)$ Для перевода из пространства скрытого состояния $h \in \mathbf{R}^h$ в пространство нашего словаря $v = |V|$ применяется линейный слой, за которым следует потенцирование и нормировка (softmax) для того, чтобы все значения были в диапазоне от 0 до 1:

$$\mathbf{s}_t = softmax(A \cdot h_t + a),$$

$$A \in \mathbf{R}^{h \times v},$$

$$b \in \mathbf{R}^v,$$

$$softmax(x_i) = \frac{\exp x_i}{\sum_{j=1}^{n} \exp x_j}$$

Эти значения в дальнейшем интерпретируются как вероятности, в частности, из них можно делать выборку с помощью различных распределений (например, мультиномиального) или брать argmax. Ниже будет подробнее рассмотрено, как процедура получения выборки (сэмплирование, sampling) влияет на результат работы модели.

Опишем алгоритм обучения такой модели [18]. Введем предварительные обозначения:

**C** – наш корпус в виде последовательности токенов после предварительной обработки. **O** – наш алгоритм оптимизации, выполняющий оптимизацию методами, основанными на методе градиентного спуска.

**Алгоритм 1** Прямой проход

```
procedure PassOneSequence
    counter ← counter + 1;
    t_0 ← n · counter;
    x_{t_0...t_0+n} ← C_{t_0...t_0+n};
    h ← h_0;
    L ← 0;
    for t ∈ {t_0 ... t_0 + n − 1} do
        e_t ← Embedding(x_t)
        h_t ← RNN(h_{t−1}, e_t, ···)
        y_t ← Output(h_t)
        L ← L + loss(y_t, D(x_{t+1}))
    BackwardPass
    for v ∈ VariableScope do
        ComputeGradient(v)
    return L
```

Для ускорения процесса расчета (особенно на графических ускорителях, GPU) а также для борьбы с чрезмерно высоким шумом при обучении по 1 примеру вычисления проводятся параллельно для небольшого набора данных, называемого минибатчем. В это случае Алгоритм 1 просто выполняется в параллельном режиме, а данные берутся последовательно из случайно перемешанного набора исходных данных. Стоит отметить, что работа в минибатч-режиме дает несмещенную оценку градиента только для 1 эпохи [18].

Алгоритм 2 опирается на процедуру Sample, которая получает на вход вектор чисел с последнего слоя сети и выдает одно слово, которое взято из словаря в соответствии с неким вероятностным распределением, параметрами которого могут выступать ее входные данные.

В литературе рассматриваются несколько вариантов этой процедуры:

1. Взятие максимума. Выходными данными является слово, значение для которого во входном векторе было наибольшим.

2. Мультиномиальное распределение. Выходные данные – выборка размера 1 из мультиномиального распределения, веса для которого получены потенцированием входных данных. Потенцирование необходимо потому, что некоторые числа могли получиться отрицательными, что недопустимо.

**Алгоритм 2** Генерация текста в авторегрессивном режиме

1: $counter \leftarrow counter + 1$;
2: $t_0 \leftarrow n \cdot counter$;
3: $x_0 \leftarrow$ <EOS>;
4: $h \leftarrow h_0$;
5: $\boldsymbol{y} \leftarrow []$
6: $\boldsymbol{x} \leftarrow [x_0]$
7: **for** $t \in \{1 \dots n\}$ **do**
8: $\quad e_t \leftarrow \text{Embedding}(\boldsymbol{x}[-1])$
9: $\quad h_t \leftarrow \text{RNN}(h_{t-1}, e_t, \cdots)$
10: $\quad y_t \leftarrow \text{Output}(h_t)$
11: $\quad o_t \leftarrow \text{Sample}(y_t)$
12: $\quad \boldsymbol{y} \leftarrow \boldsymbol{y} + o_t$
13: $\quad \boldsymbol{x} \leftarrow \boldsymbol{x} + \mathbf{D}(o_t)$
**return** $\boldsymbol{y}$

### 1.2.4 Рекуррентный слой

Большая часть вычислений в языковой модели состоит в перемножении *одних и тех же* матриц на все новые и новые входные данные и скрытые состояния. Несмотря на то что это обеспечивает запоминание моделью временных зависимостей, это представляет собой серьезное затруднение для языковых моделей. Рассмотрим нашу самую простую модель в качестве примера:

$$h_1 = \sigma(Wx_0 + Uh_0 + b)$$

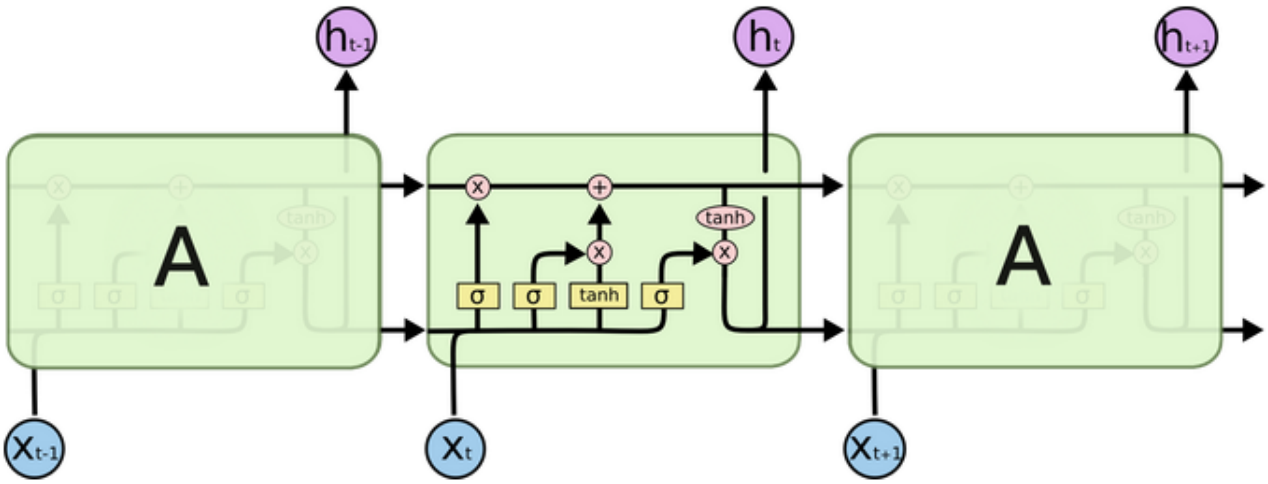
$$f(x_2, y_2) = \sigma(Wx_1 + Uh_1 + b) = \sigma(Wx_1 + U \cdot \sigma(Wx_0 + Uh_0 + b + b)$$

Можно видеть, что во время обучения мы последовательно выполняем большое количество умножений *на одни и те же матрицы*, за некоторыми из которых следует нелинейное преобразование (в нашем случае сигмоида) и которые в итоге влияют на скрытое состояние модели *независимо* от входных данных. Данную проблему можно назвать неконтролируемым изменением информации (information morphing), которое приводит к тому, что при достаточно большом числе шагов вследствие многократного умножения сама на себя матрица $W^n$ начинает определяться в большей степени ее собственными числами $\lambda_i, i \in 1 \cdots n$, тем самым в случае $\lambda_i < 0$ соответствующая строка будет иметь малые значения, а в случае $\lambda_i > 0$ – большие. Данные значения, в свою очередь, окажут влияние на градиенты всех параметров при обратном распространении ошибки, тем самым процесс обучения перестанет быть эффективным.

В связи с этим данной работе используется LSTM (Long Short-Term Memory, долгая краткосрочная память) [21], и GRU (Gated Recurrent Unit, [11]), поскольку в них произведены изменения в вычислительном графе, позволяющие эффективно сохранять информацию на протяжении десятков и сотен шагов: механизм гейтов – специальных функций, вычисляющих, как прочитанный на каждом временном шаге символ должен повлиять на сохраненное скрытое состояние и на выходные данные. На каждом временном шаге сначала вычисляются эти значения (input, forget, output gate для LSTM, update and reset gate для GRU), а потом происходит обновление скрытого состояния и формирование выходного вектора.

$$i_t = \sigma(x_t U^i + h_{t-1} W^i + b_i)$$

$$f_t = \sigma(x_t U^f + h_{t-1} W^f + b_f)$$




$$o_t = \sigma(x_t U^o + h_{t-1} W^o + b_o)$$

$$c_t = \tanh(x_t U^g + h_{t-1} W^g + b_g)$$

$$C_t = \sigma(f_t \cdot c_{t-1} + i_t \cdot c_t)$$

$$h_t = \tanh(c_t) \cdot o_t$$

$i$ – входной шлюз,
$f$ – шлюз забывания,
$o$ – выходной шлюз,
$c$ – кандидатское состояние памяти,
$C$ – состояние памяти,
$h$ – выдаваемое значение

Здесь $a \cdot b$ обозначает поэлементное умножение, $ab$ – матричное умножение.

Глядя на формулы, можно заключить, что в LSTM решена как проблема неконтролируемого изменения скрытого состояния, так и проблема его ограниченности в течение работы модели и проблема координации чтения и записи в память, которая могла бы привести к тому, что случайные действия еще не обученной модели в более ранние моменты времени приводили бы к серьезным изменениям в более поздние моменты времени, что негативно бы сказалось на процессе обучения.

**Ускорение с учителем**

Несмотря на вышеописанные средства, применяющиеся для улучшения работоспособности таких моделей, рекуррентные нейронные сети представляют особую сложность при тренировке по следующим причинам:

1. Общие параметры во многих частях вычислительного графа
   В связи с этим наиболее распространенным способом обучения таких моделей является так называемое Ускорение с учителем (Teacher Forcing) – подход, в котором во время обучения на каждом следующем шаге модель получает на вход не результат ее работы, полученный сэмплированием из предыдущего шага, а правильный токен. Это позволяет существенно ускорить модель, но во время работы, когда правильный ответ неизвестен, динамика скрытого состояния может стать непредсказуемой [31].

2. Непредсказуемая динамика скрытого состояния во время вывода
   [31] обращает внимание на то, что большая выразительная способность глубоких моделей может

принимать плохой оборот во время самостоятельной работы: независимо от использования Teacher Forcing'а оказывается, что скрытое состояние модели может оказаться в области, где оно никогда не оказывалось во время обучения, тем самым модель из режима интерполяции между известными точками переходит в режим экстраполяции в неизвестной ей области.

Следующие подходы предложены для борьбы с вышеперечисленными проблемами:

1. Изменение в вычислительном графе [19], [21].
2. Использование обучения с подкреплением для того, чтобы модель могла исследовать свое пространство параметров [32].
3. Использование Состязательного обучения [31].

# Нейронный машинный перевод

## 2.1 Статистический машинный перевод

С увеличением вычислительных мощностей (и общечеловеческим прогрессом) появилась возможность решать задачу устранения ошибок как задачу машинного перевода. Для этого использовались системы статистического машинного перевода (СМП): [24].

Основной идеей основанного на фразах машинного перевода является вычисление следующих признаков: выравнивание (alignment), плодоносность (fertility) и изменение порядка (reordering).

#### Выравнивание

Выравнивание – процедура, соотносящая одному или нескольким словам в предложении из одного языка одно или несколько слов в предложении из другого языка. Рассмотрим пример:
Кошка ходит по окну. (рус) – The cat is walking on the window (англ) Так как в русском языке нет артиклей, то вся именная фраза "the cat" должна быть выровнена со словом "кошка". Аналогично со вспомогательным глаголом – в русском языке их нет, поэтому вся глагольная фраза "is walking" должна быть выровнена с одним глаголом "ходит".

#### Плодоносность

Плодоносность – процедура, сопоставляющая каждому токену из исходного предложения число токенов в целевом предложении, которые получаются с его помощью. Она во многом аналогична вниманию, только если первое является биективным отображением, то плодоносность – только инъективно. На этом же этапе с помощью полученного ранее внимания производится генерация одного или нескольких слов для каждого слова из исходного предложения.

#### Переупорядочивание

Переупорядочивание производится с помощью статистической языковой модели, его задача – сделать итоговое предложение максимально "естественно звучащим" (fluent), используя большую языковую модель в качестве метрики качества. Ниже представлена примерная схема работы статистического машинного переводчика.

#### Недостатки СМП

Статистический машинный перевод, показывая сравнительно хорошие результаты на простых предложениях, испытывает трудности при работе с длинными предложениями с богатой синтаксической структурой. Причина тому – необходимость выполнять сложное выравнивание и сглаживание, что хорошо видно при рассмотрении следующих примеров:

Я позвоню господину Фишеру. (рус) – Ich rufe Herrn Fischer an. (нем) В этом примере an – отделяемая приставка, и начальная форма глагола anrufen (звонить), однако согласно правилам языка она должна

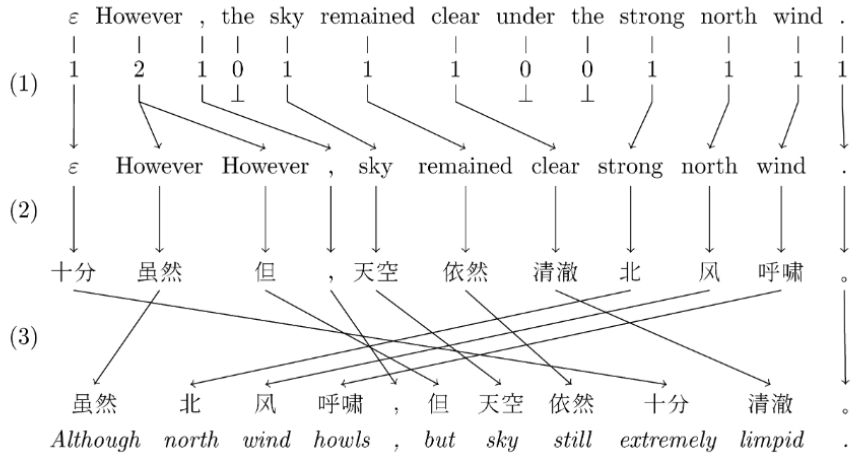



Рисунок 3: Перевод предложения статистическим машинным переводом [24]

находиться в конце предложения. Таким образом, слово "позвоню" должно быть выровнено с двумя (в случае более сложного времени больше) словами, которые находятся не подряд. Ситуации, когда синтаксические правила в двух языках диктуют очень сложные выравнивания, и являются сложными для статистического машинного перевода. Еще один пример – перевод с английского на французский [1]:

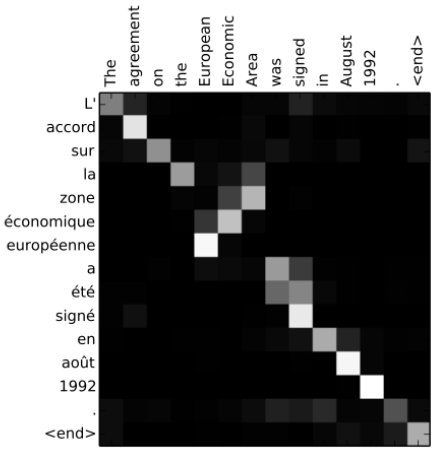


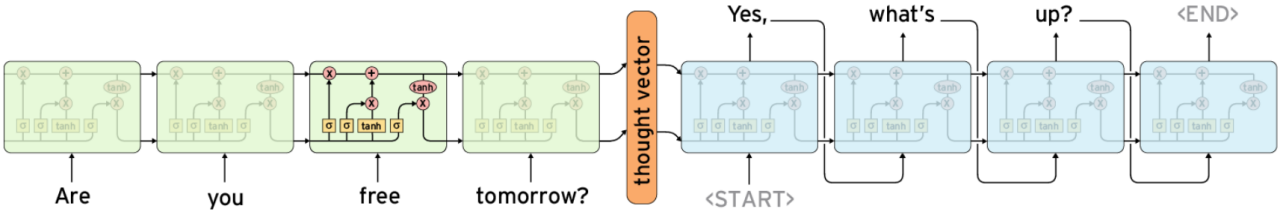

Рисунок 4: Пример выравнивания в англо-французской языковой паре, которое человек счел бы корректным

Несмотря на различные методы решения вышеописанных проблем (например, обучение механизмов выравнивания в unsupervised режиме [16]), а также использования разнообразных методов моделирования языка в точках, отдаленных от точек в обучающем корпусе, задача во многом оставалась нерешенной.

## 2.2 Кодировщик-декодировщик

Системы нейронного машинного перевода состоят из двух основных компонент: кодировщика (encoder) и декодировщика (decoder). Их названия отражают их функциональные особенности: первый из них получает на вход предложение на исходном языке и на выходе отдает т. н. "код" – представление предложения в пространстве конечной наперед заданной размерности. Задача же декодировщика – расшифровать "код" в текст – последовательность токенов на целевом языке. Поскольку пространство

"кодов" конечной длины, то оба компонента системы вынуждены эффективно сжимать информацию, тем самым обеспечиваю вычислительную достижимость всего алгоритма.
Традиционно и для кодировщика, и для декодировщика использовались рекуррентные сети, однако в последнее время получили распространение сверточные сети [?] или сети без рекурсии, в которых темпоральные зависимости выражены в виде добавления информации о временном шаге к каждому токену, а вычисления проводятся с помощью самовнимания (self-attention) [58].




## 2.3 Внимание

Описанная выше система обладает, однако, существенным недостатком, вытекающим из того, что в "код" нужно уместить всю необходимую для расшифровки информацию. Несмотря на то что теоретически это выполнимо, практическая оптимизация в высокомерном пространстве приводит к тому, что при длинах входной последовательности порядка десятков "код" перестает вмещать в себя необходимую информацию, тем самым алгоритм становится бесполезным.
Для устранения данного информационного горлышка (information bottleneck) был предложен механизм внимания [?], заимствованный из статистического машинного перевода, где он назывался выравнивание (alignment). Суть этого механизма в следующем: кодировщик выдает не только "код" последовательности, но и все скрытые состояния, полученные в ходе применения рекуррентной ячейки на каждом своем шаге, а декодировщик, в свою очередь, на каждом шаге вычисляет важность каждого скрытого состояния кодировщика для данного момента времени, после чего берет взвешенную сумму всех скрытых состояний для данного момента времени и использует ее во время своих расчетов.

$$\alpha_j = \frac{\exp s_j}{\sum_{k=1}^{T} \exp s_k}$$

$$d_i = \sum_{1}^{T} \alpha_j \cdot h_j$$

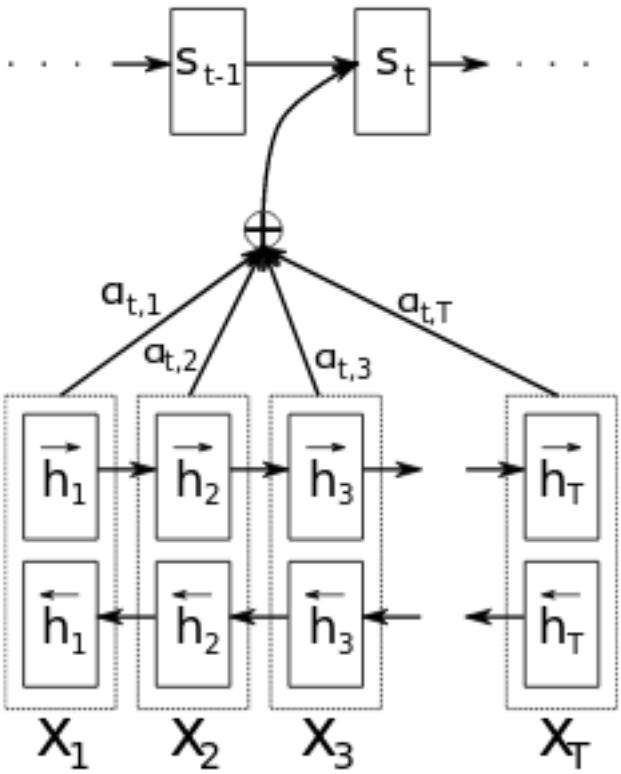
Существует несколько видов вычисления важностей:

1. Аддитивное
$$s_{i,j} = f(A \cdot x_i + B \cdot h_j),$$

2. Мультипликативное
$$s_{i,j} = f(x_i^T \cdot W \cdot h_j),$$

3. Непараметрическое
$$s_{i,j} = f(x_i \cdot h_j)^T$$

Стоит отметить, что данный подход (называемый иначе "наивное внимание", чтобы была возможность отличить его от памяти) хорошо работает только тогда, когда удается обучить механизм выравнивания, для чего требуется определенная плотность обучающих данных для всех рассматриваемых длин последовательно




Однако, последовательности больших длин оказываются сильно разреженными, поэтому данный механизм в данной реализации применяют на уроне предложения (или нескольких предложений), но не длинных текстов.

### 2.3.1 Функции потерь

Выше мы уже рассмотрели метрики, используемые для оценки качества моделей, однако не все они в явном виде применимы для нашей задачи.

#### Недифференцируемые

Связано это с тем, что многие из них (метрики, основанные на edit distance, а также BLEU и GLEU) недифференцируемы, что приводит к невозможности обучать модели с помощью методов, использующих градиентный спуск. Одной альтернативной является использование так называемых градиентов политик (policy gradients) и алгоритма REINFORCE, однако данный подход обладает важными недостатками:

1. Долгое время обучения и медленная сходимость.
   В алгоритме REINFORCE делается оценка "градиентов" по нескольким вычислениям модели, и обычно при числах меньших 10-20 дисперсия оказывается слишком большой, чтобы получаемая оценка градиентов позволяла модели хоть насколько-то быстро сойтись

2. Необходимость решения-бейзлайна.
   Даже при большом числе шагов дисперсия все еще оказывается достаточно заметной, и для ее уменьшения необходимо использовать специальный алгоритм, называемый бейзлайном, который тоже, подобно нашему основному алгоритму, решает исходную задачу. При этом качество бейзлайна напрямую влияет на дисперсию, особенно на конечных этапах сходимости [57].

Несмотря на это, отдельные авторы отмечают улучшение результатов на тех или иных датасетах [54]. Мы же оставляем использование недифференцируемых метрик на будущее.

**Дифференцируемые**

Поскольку наш алфавит (будь то буквы или символы) конечен и дискретен, то категориальная кросс-энтропия является наилучшим выбором для нашей задачи. Эта же метрика используется при обучении во многих других задачах, таких как автоматическое моделирование языка, машинный перевод или автоматическое реферирование. Данная метрика вычисляется следующим образом:

$$L(s, r) = \sum_{i=1}^{n_{max}} l(s_i, r_i),$$

где

$$\mathrm{l}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right),$$

$s$ – вероятности за каждый класс, $r$ – номер правильного класса, $n_{max}$ – максимальная из длин предложений (при необходимости более короткое предложение необходимо дополнить специальным <PAD> токеном, чтобы его длина совпала с длиной более длинного предложения.

Благодаря тому, что данная метрика вычисляется по отдельности для каждого слова, вычисления могут быть эффективно распараллелены, кроме того, она хорошо подходит для использования вместе с ускорением с учителем.

### 2.3.2 Отбор гипотез

Многие задачи обработки естественного языка допускают неоднозначность в выходных данных, например, ‘Hello!‘ можно перевести как ‘Добрый день!‘ и как ‘Здравствуйте!‘. Таким образом, встает необходимость использования нескольких гипотез в алгоритме 2.

Вторая причина хранить и динамически обновлять несколько гипотез состоит в том, что наиболее правдоподобная на начальных шагах гипотеза может оказаться не самой правдоподобной по окончании, что приводит к субоптимальному ответу однако последовательного принятия решений в языковой модели.

Для решения обеих этих проблем предложен алгоритм лучевого поиска (Beam Search), в котором для каждой гипотезы (луча, beam) вычисляются наиболее вероятные продолжения, после чего из всей совокупности гипотез-продолжений выбирается $k$ наиболее правдоподобных, где $k$ – это ширина поиска (число лучей). На начальном этапе все гипотезы инициализируются одинаковым StartOfSentence токеном.

**Алгоритм 3** Лучевой поиск

```
procedure STEP
    x_t ← E(w_t);
    B ← [[SOS], [SOS], ...]
    counter ← counter + 1;
    t_0 ← n · counter;
    x_0 ← <EOS>;
    h ← h_0;
    for t ∈ {1 ... n} do
        for b ∈ B do
            e_t^b ← Embedding(x[−1])
            h_t^b ← RNN(h_{t−1}, e_t, ···)
            y_t^b ← Output(h_t)
            b ← b + y_t^b
    B = topk(B)
    return y
```

На практике $k$ выбирается не очень большим (4-10) для нейронных методов и 20-50 для статистических методов. Данный алгоритм в $k$ раз медленнее алгоритма 2, поскольку на каждом временном шаге необходимо производить вычисления для каждого из лучей по отдельности, к тому же он требовательнее по памяти, но ненамного (в масштабах всей модели).

Можно видеть, что данный алгоритм тоже "жадный": может получиться так, что все $k$ наших вариантов субоптимальны. Кроме того, внутри него применяется недифференцируемая операция argmax, что влечет дополнительные сложности в системах, тренируемых методами на основе градиентного спуска. Существуют дифференцируемые [59] и менее жадные [32] модификации данного алгоритма, применение которых остается за рамками нашей работы.

# Эксперимент

## 3.1 Предварительное исследование: синтетический язык

### 3.1.1 Регулярные языки

Рассмотрим грамматику $(abcabcd)^*$. Можно видеть, что принимающий ее конечный автомат имеет 7 состояний. Данная грамматика интересна для экспериментов, потому что скрытые состояния сети возможно проинтерпретировать.

В качестве модели возьмем LSTM-сеть, каждый символ закодируем индивидуальным кодированием (one-hot encoding) – сопоставим $a_i$ вектор $(0, \ldots, 0, 1, 0, \ldots, 0)$, где 1 стоит на $i$-й позиции, а длина вектора – размер словаря. Размер скрытого слоя возьмем равным 2, а выходной слой – линейное преобразование в наш алфавит из 4 символов.

Обучим нейронную языковую модель этой грамматике. Можно видеть, что скрытые состояния сгруппированы в кластеры как во время работы в авторегрессивном режиме, так и во время работы в неавторегрессивном режиме.

Однако при наличии шума скрытое состояние модели серьезно изменяется, как изменяются и вероятности выдачи правильного (в незашумленной версии) символа. Помимо этого, претерпевают серьезные изменения распределения вероятностей на следующий символ: если при работе в незашумленном режиме распределение всегда имеет четкий максимум, то при наличии шума максимум размывается, и дисперсия распределения существенно уменьшается, а само распределение становится ближе к равномерному.

Деградировавшее скрытое состояние серьезно влияет на оценки вероятностей на следующее слово: распределение становится гораздо ближе к равномерному, а дисперсия становится ниже.

## 3.2 Предварительное исследование: английский язык

Чтобы оценить способность статистических и нейронных моделей к анализу ошибок, проведем предварительное исследование, используя датасет FCE. Корпус был подвергнут следующей предобработке:

1. Было произведено искусственное увеличение размера корпуса: если в предложении $k$ ошибок, то *вместо* ошибочного варианта рассматривались $k$ ошибочных вариантов, каждый из которых содержал ошибку на $k$-м месте. Каждый из этих примеров сравнивался с полностью корректным вариантов.

2. Не рассматривались предложения, содержащие ошибки в двух и более последовательно идущих словах. Таких предложений оказалось 3269 из 4124, то есть около 80%.

### 3.2.1 Статистическая языковая модель

Языковая модель на 4-граммах с back-off'ом [20] обучалась на корпусе свободно доступных английских текстов из библиотеки Гутенберга, находящихся в открытом доступе. Объем корпуса составил 1 млрд токенов, были исключены все неанглийские слова, все слова были приведены к нижнему регистру, пунктуация

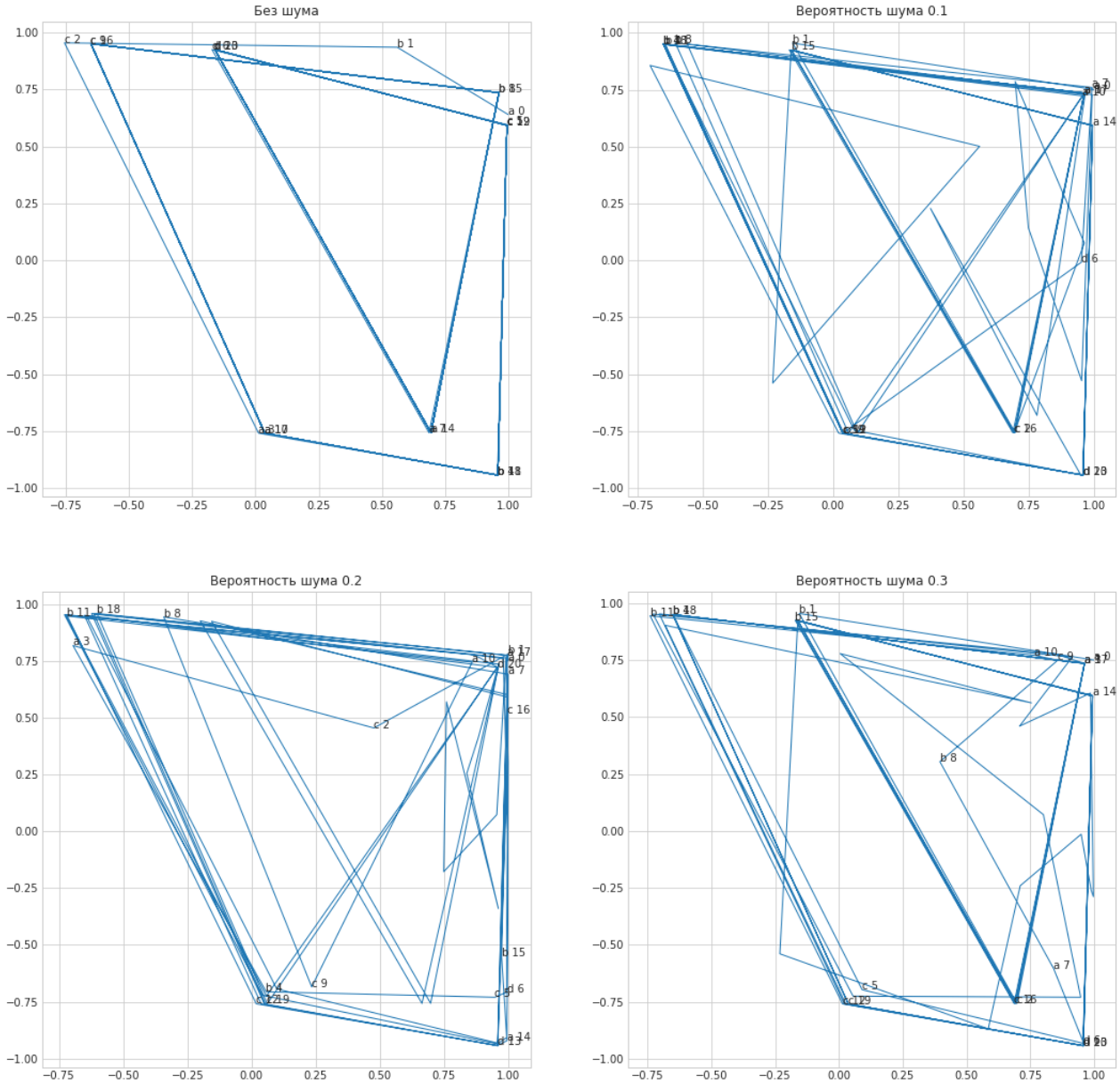



Рисунок 5: Примеры динамики скрытого состояния в языковой модели, обученной на регулярном языке. На всех изображениях по осям абсцисс и ординат отложены скрытые состояние первого и второго рекуррентных нейронов соответственно.

отделена пробелами с обеих сторон. Полученная языковая модель использовалась для вычисления вероятностей n-грамм в корпусе FCE для правильного и ошибочного предложений.

**Гипотеза**

Ошибочные n-граммы будут иметь меньшую вероятность, чем их исправления, поскольку вероятность встретить последние в обучающем корпусе выше.
Данная гипотеза верна для модельных примеров, рассмотренных выше (например, 3-грамма "(])" невероятна в грамматике правильных скобочных выражений).

**Эксперимент**

Статистическая языковая модель показала следующие результаты:

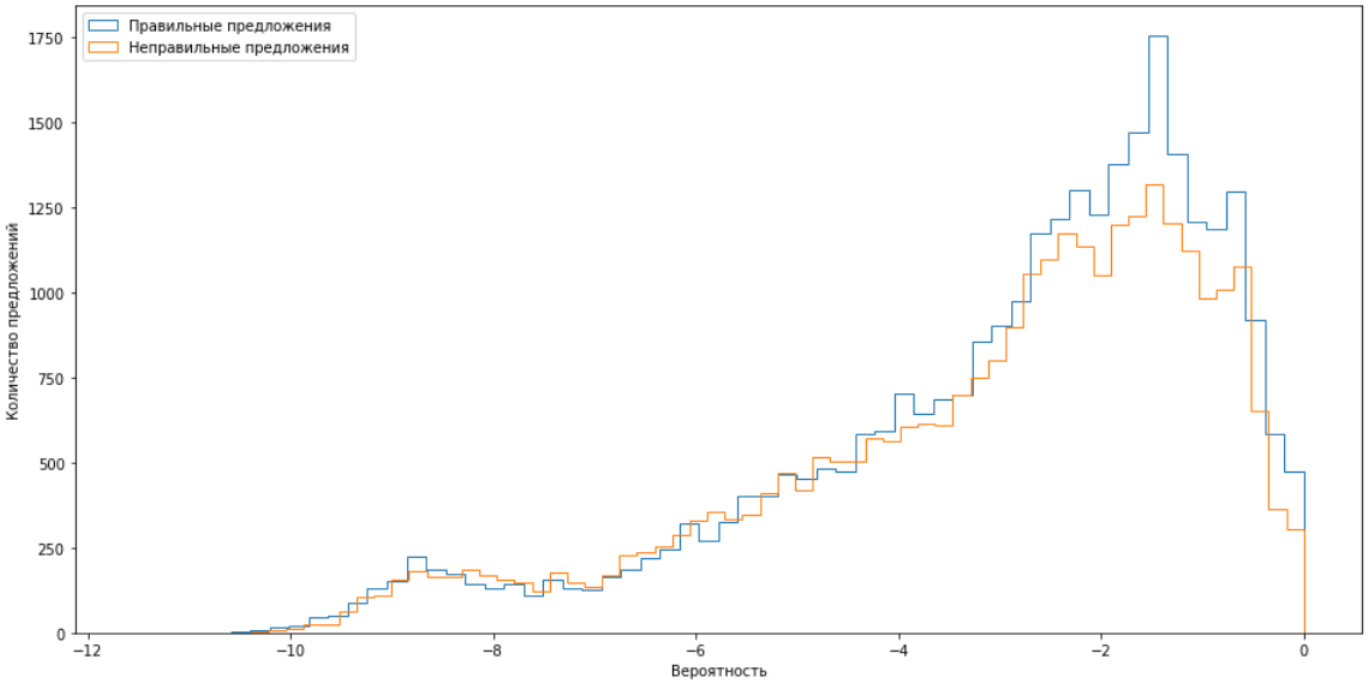



Рисунок 6: Гистограмма распределений вероятностей для корпуса FCE

Статистический критерий Уилкоксона говорит, что гипотеза "$n$-граммы, соответствующие грамматически некорректным предложениям, имеют такую же вероятность, что и их исправленные альтернативы", **принимается**.

Проведем эксперимент в аналогичной постановке: языковые модели, обученные на копусе Penn Treebank, работают в авторегрессивном режиме. На рисунках представлены дисперсии ненормированных оценок на следующее слово в зависимости от токена в данный момент времени для двух моделей: в первой использован словарь размера 10000, во второй – словарь буквенных $n$-грамм размера 11677. Можно видеть, что в обоих случаях дисперсия при обработке моделью UNK токена оказывается меньшей, чем при обработке других слов. Отдельный интерес представляют артикли, потому что они используются в речи достаточно часто, в то время как правила их применения достаточно сложно формализовать.

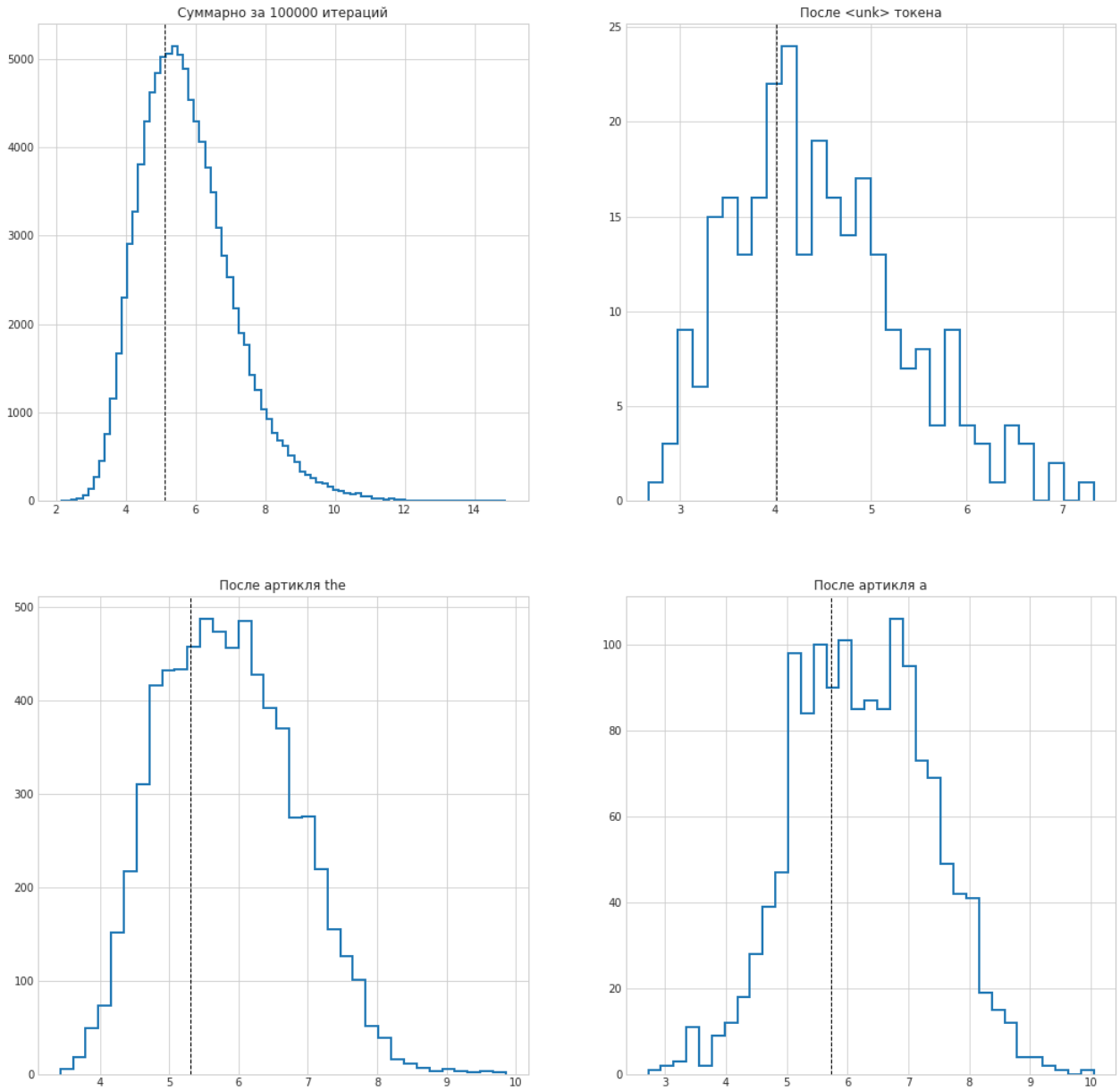



Рисунок 7: Гистограмма распределения дисперсий ненормированных оценок на следующее слово для языковой модели, обученной на Penn Treebank. На всех графиках по оси абсцисс отложено значение дисперсии, по оси ординат – количество измерений. Штрихованная черная линия соответствует медиане распределений.

### 3.2.2 Нейронная языковая модель

Нейронная языковая модель обучалась по алгоритму 1, а валидировалась по алгоритму 2 на валидационной части корпуса enwiki-8. Корпус был предобработан с помощью токенизации и перевода в нижний регистр. Рассматривались модели, обучавшиеся на уровне слов или на уровне byte-pair-encoding (алгоритм 3).

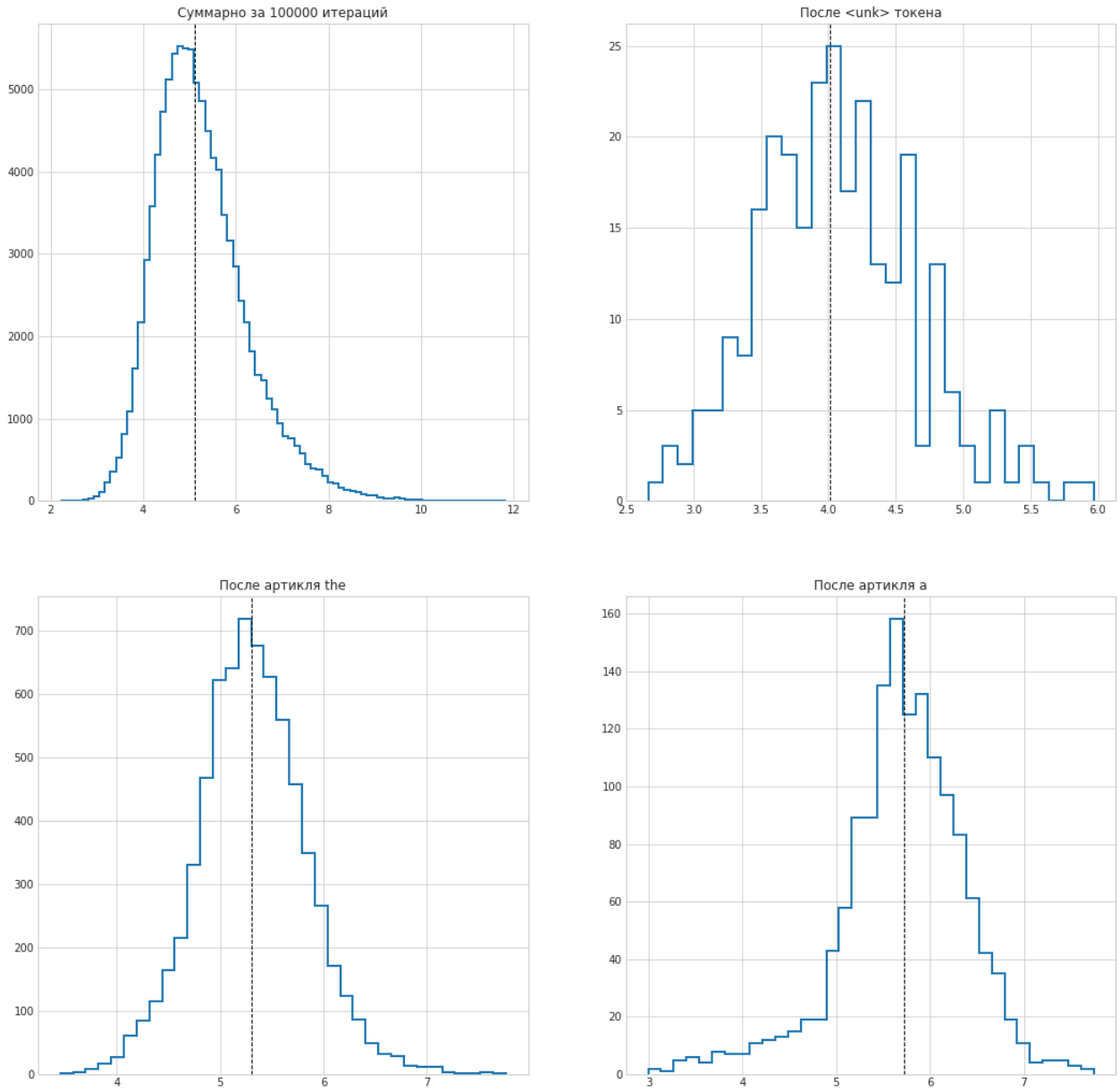



Рисунок 8: Гистограмма распределения дисперсий ненормированных оценок на следующее слово для языковой модели, обученной на Penn Treebank. На всех графиках по оси абсцисс отложено значение дисперсии, по оси ординат – количество измерений. Штрихованная черная линия соответствует медиане распределений.

В 841 случае из 1215 логарифм отношения оказался положительным, что показывает, что информация на уровне морфем играет важное значение для обработки ошибок. Тем не менее, гипотеза о том, что

| Тип словаря | Размер словаря | Размер скрытого слоя | $p$-value |
|---|---|---|---|
| слова | 30000 | 512 | 0.06 |
| BPE | 10000 | 512 | 0.12 |

Таблица 2: Результаты работы нейронных языковых моделей

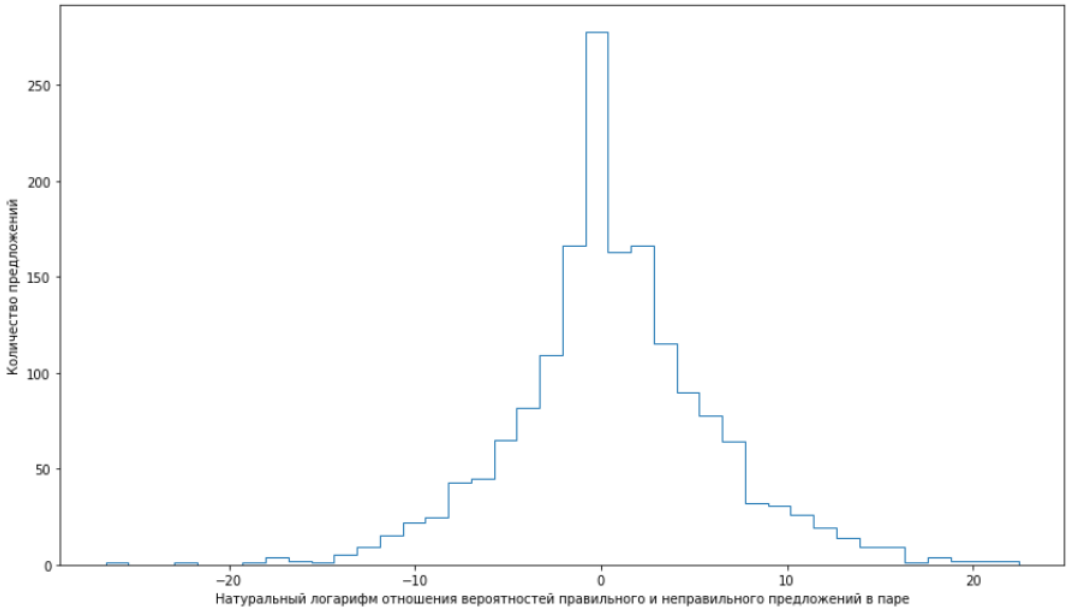



Рисунок 9: Гистограмма распределений отношения вероятностей для модели на уровне слов

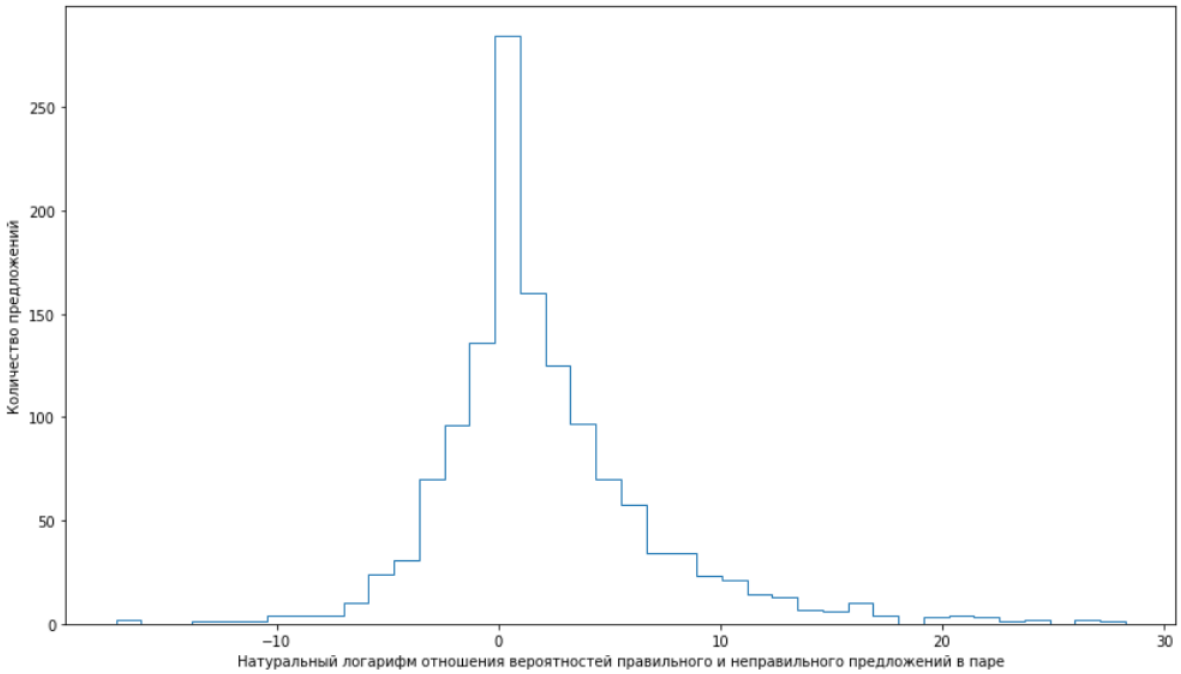



Рисунок 10: Гистограмма распределений отношения вероятностей для модели на уровне n-грамм

**Качественный анализ**

Посмотрим на примерах выдачу модели (взята модель на уровне BPE с 512 нейронами в скрытом слое).

| Предложение | Исправленное предложение | Отношение вероятностей |
|---|---|---|
| i is going to the kitchen . | i am going to the kitchen . | 512 |
| i was shocked because i had already spoken with them and i had got two autographs. . | i was shocked because i had already spoken with them and i had got two autographs. . | 0.005 |
| i was wondering if you could give me some lesson while i am there. | i i was wondering if you could give me some lessons while i am there. | 760.67 |
| i am keen on photography so i would love to participate in this activity . i am very good at it and it would be a pleasure for me to help the others. | i am keen on photography so i would love to participated at this activity , i am very good at that and it would be a pleasure for me to help the others. | 1492020.22 |
| thanks for the kind letter you send me las week . | thanks for the kind letter you sent me last week . | 1157645388.63 |
| i think also housewifes will receive more help with a robotor in the household | i think also housewives will receive more help with a robot in the home | 0.00003 |

Таблица 3: Выдача модели (тексты подвергнуты токенизации, орфография и пунктуация сохранены)

| Параметр | Значение |
|---|---|
| Оптимизатор | Adam |
| Эмбеддинги для энкодера и декодера | Фиксированы |
| Количество гипотез | 5 |
| Размер минибатча | 30 |

Таблица 4: Параметры обучения, остававшиеся неизменными.

## 3.3 Нейронный машинный перевод

Модели на основе нейронного машинного перевода обучались с использованием Алгоритмов 1 – 4. Использовалас библиотека OpenNMT для фреймворка PyTorch [49]. В таблице ниже приведена информация о параметрах не менявшихся в течение обучения.

Английская часть корпуса Lang-8 была подвергнута следующей предобработке:

1. Удаление всех символов с ascii-кодами большими 128 или не из таблицы ascii
2. Перевод в нижний регистр
3. Разбиение по предложениям
4. Токенизация

Обучение производилось на графическом ускорителе NVIDIA GeForce 1080 X ti.

### 3.3.1 Количественная и качественная оценка

Рассмотрим на примерах, как работает наш "переводчик". Использована модель с BPE-256-512

Можно видеть, что модель справляется с исправлениями *простых* грамматических ошибок даже тогда, когда сталкивается со словом не из своего словаря.

| Параметр | Значение |
|---|---|
| Оптимизатор | Adam [18] |
| Скорость обучения | 0.0001 |
| Эмбеддинги для энкодера и декодера | Общие |
| Количество гипотез | 5 |
| Размер минибатча | 30 |
| Тип энкодера | BiLSTM |
| Тип декодера | LSTM |

Таблица 5: Фиксированные параметры

| token type | vocab length | embeddings size | hidden size | layers | token-level acc | F0.5 |
|---|---|---|---|---|---|---|
| слова | 10000 | 256 | 512 | 1 | 92.4 | 28.15 |
| BPE | 10000 | 256 | 512 | 2 | 93.4 | 31.07 |
| BPE | 30000 | 256 | 512 | 1 | 92.6 | 27.68 |
| BPE | 30000 | 64 | 768 | 1 | 80.1 | 23.12 |
| слова | 30000 | 256 | 512 | 1 | 92.1 | 29.24 |
| BPE | 10000 | 256 | 768 | 2 | 89.8 | 28.7 |

Таблица 6: Результаты метрик в зависимости от гиперпараметров

| Исходное предложение | Вывод модели | Вывод человека |
|---|---|---|
| i going to an kitchen . | i am going to the kitchen . | i am going to the kitchen. |
| benedict cumberbatch am running to store . | benedict cumberbatch is running to the store . | benedict cumberbatch is running to the store . |
| he have gat one models . | he has got one models . | he has got one model . |
| i likes apples bananas and a coffee . | i like apples, bananas and coffee | i like apples and a coffee |
| important deadlines must be meet . | important deadlines must be met . | important deadlines must be met . |
| he is the more clever guy . | he is a clever guy . | he is the cleverest guy . |

Таблица 7: Выдача модели (тексты подвергнуты токенизации, орфография и пунктуация сохранены)

Одной из проблем глубоких нейронных сетей является их неинтерпретируемость ввиду большого количества параметров и сложно протекающего процесса оптимизации. Пролить свет на происходящее помогают "карты внимания": двумерная матрица, строки которой отвечают за выдачу модели, столбцы – за начальные слова, а значения – за коэффициенты $\alpha_j$.

Ниже приведены примеры таких карт для нескольких предложений:

Рисунок 11: Карта внимания

## 3.4 Перенос знаний

Перенос знаний (Transfer Learning) [64] – название подхода, при котором модели, обученные на одних задачах, используются при обучении на других задачах. такая технология особенно удобна при работе с нейронными сетями, поскольку появляется простая возможность переносить и фиксировать параметры моделей во время обучения.

Данный подход дает следующие преимущества:

1. Меньшее время обучения даже в тех случаях, когда параметры не фиксированы, поскольку стартовая точка для оптимизации модели уже достаточно удачна

2. Меньшее количество данных для сходимости новой модели.

### 3.4.1 Текущее применение в обработке естественного языка

В системах компьютерного зрения Transfer Learning уже занял важное место: при дообучении многих моделей фиксированы параметры сверточной компоненты сети, например ImageNet [30], таким образом, удается сэкономить до 90% всех вычислений и использовать датасеты на порядки меньших размеров.

В настоящее время активно ведутся работы по применению данной технологии в обработке естественного языка, что во многом вызвано отсутствием большого количества данных для многих языков. В частности,

Рисунок 12: Карта внимания

[64] сообщают о существенном росте BLEU при работе с низкоресурсными языковыми парами, такими как англо-узбекский или англо-турецкий. В [53] рассматривают задачи классификации и сообщают об успешном дообучении моделей на выборках в 100 раз меньшего объема. Наконец, в [**?**] тренируют модель для генерации вопросов и ответов на них одновременно, добиваясь высоких результатов на датасете SQUAD [51]. Серьезных результатов удается добиться, обучая модель сразу на нескольких языках [59], в этом случае возможно так-называемое "обучение без данных", когда машинный перевод из языка X на язык Y удается осуществить, не имея параллельного корпуса этих языков, но имея параллельный корпус X–Z и Y–Z.

Мы рассматриваем перенос знаний со следующей перспективы: имея модель, обученную на большем количестве данных (модель-родитель), получить статистически значимый результат на меньших данных (модель-ребенок), используя обученные веса модели-родителя в качестве стартовых значений для процедуры оптимизации.

### 3.4.2 Результаты

В таблице ниже приведено сравнение моделей с переносом знаний и бейзлайнами (моделями, обучавшимися заново)

### 3.4.3 Качественная оценка и обсуждение

Можно видеть, что модель выучила определенные закономерности:

Однако, наряду с не очень высокими (по сравнению с людьми) результатами метрик, при качественном анализе выявлены следующие недостатки:

1. Исправленные предложения часто получаются короче, чем исходные: причастные или деепричастные обороты часто обрезаются, в то время как в них обычно содержится не более одной ошибки.

| init type | token type | vocab length | embeddings size | hidden size | layers | token-level acc | F0.5 |
|---|---|---|---|---|---|---|---|
| en | слова | 10000 | 256 | 512 | 1 | 82.6 | 21.35 |
| en | BPE | 10000 | 256 | 512 | 2 | 80.7 | 20.15 |
| en | BPE | 30000 | 256 | 512 | 1 | 84.1 | 23.43 |
| random | BPE | 30000 | 64 | 768 | 1 | 72.1 | 15.4 |
| random | слова | 50000 | 128 | 512 | 1 | 65.2 | 17.12 |
| random | BPE | 10000 | 256 | 768 | 2 | 64.3 | 14.89 |

Таблица 8: Результаты метрик

Рисунок 13: Карта внимания для предложения "Geht mich mir bitte."

2. Не всегда исправляются повторения: например, предложение "A man wearing a suit and tie and a tie." содержит повторение, которое модель не убирает ни в одной из гипотез.

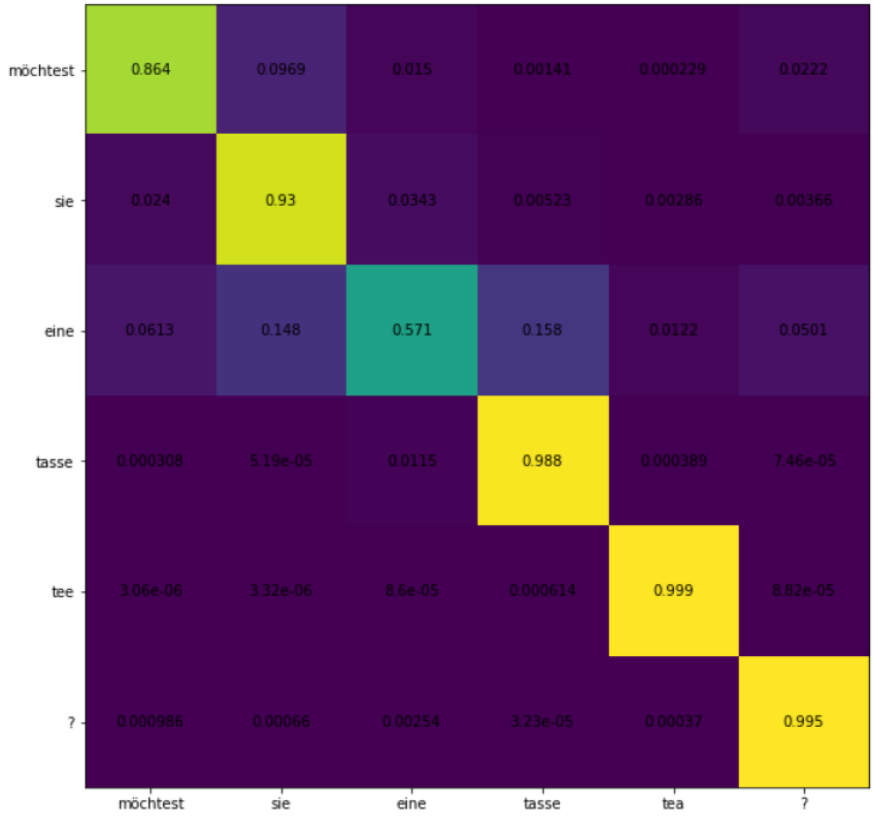



Рисунок 14: Карта внимания для предложения "Möchtest sie eine tasse tea?"

## 3.5 Решение задачи классификации

Для языковых приложений может быть полезным расширенная справка, выдаваемая пользователю после того, как он совершил ошибку. В нашем случае необходимо дополнительно провести классификацию по типам из корпуса NUCLE, выполняемую по следующему алгоритму:

1. Выполнить парсинг исправленного предложения с помощью статистического или нейронного парсера (мы используем SpaCy).

2. Для каждого слова в исправленном предложении, если оно отсутствовало в исходном предложении или если его позиция изменилась, на основе правил, составленных с помощью извлекаемых парсером лингвистических признаков, таких как часть речи или тип синтаксической зависимости, отобразить информацию о слове, а так же найденном типе ошибки.

Стоит отметить, что выполнять парсинг исходного предложения не стоит, поскольку парсеры обучаются всегда на грамматически корректных данных, поэтому их использование на грамматически некорректных данных ввиду рассмотренного в 4.1 эффекта представляется сомнительным. Так, на рисунке 14 представлено дерево разбора предложения "They were good doing". Можно видеть, что неправильный порядок слов серьезно повлиял на извлеченные парсером признаки по сравнению с предложением "They were doing good".

Пример – ошибка ArtOrDet. Классифицировать ее сравнительно просто – достаточно найти слово с частью речи DET (Determiner) – в английском языке сюда входят как артикли, так и другие слов (some, any, one, ..). Если это слово попало в аннотации – значит, найдена ошибка такого вида.

В качестве второго примера рассмотрим Vform – неправильную форму глагола. Для ее поиска нам необходимо найти глагол, который присутствует в исходном тексте, но в другой форме. Например: I am go to the kitchen. – I am going to the kitchen. В данном случае оба слова (go и goes) имеют одну и ту же

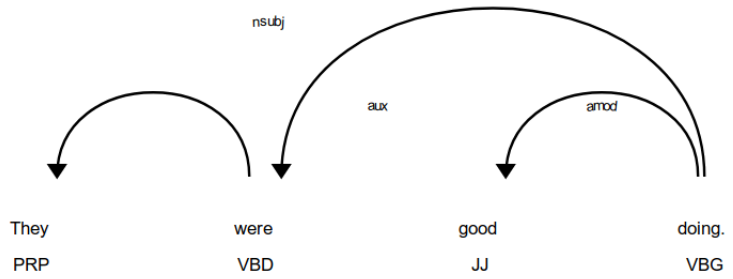



Рисунок 15: Дерево разбора для исходного предложения

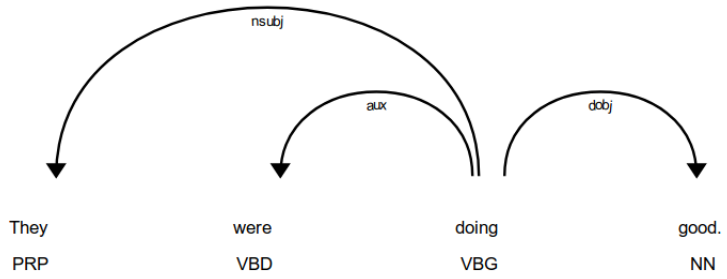



Рисунок 16: Дерево разбора для исправления, выполненного человеком

лемму и оба являются глаголами, но у них разный тег: VB у go и VBG у goes. Это дает нам возможность классифицировать наши ошибки.

# Заключение

## 4.1 Резюме

В данной работе мы рассмотрели задачу по поиску, классификации и устранению ошибок в письменной речи. Были описаны алгоритмы детекции ошибок на основе обучения с учителем и без учителя, было проведено их сравнение на корпусе размеченных английских текстов. Количественно показано преимущество моделей обучения с учителем над моделями обучения без учителя. Проведены и описаны эксперименты, позволяющие выявить недостатки моделей обучения без учителя и обосновать их ограниченное применение в рассматриваемой задаче. Проведены эксперименты по переносу языковой модели с языка на язык, показано количественное улучшение на тестовом корпусе.

## 4.2 Полученные результаты

1. Количественно и качественно показано негативное влияние шума на нейронные языковые модели, обозначены границы их применимости для задачи детекции ошибок.

2. Показано влияние конечного словаря на динамику нейронных языковых моделей и даны рекомендации по частичному устранению данной проблемы.

3. Удалось осуществить частичный перенос весов обученной языковой модели с английского языка на немецкий и получить прирост в 24 % при прочих фиксированных гиперпараметрах.

## 4.3 Дальнейшая работа

Дальнейшую работу можно разделить на две большие категории: создание более совершенных алгоритмов и применение обученной модели в новых задачах. В первой категории большое значение имеют следующие направления:

1. Обучение, использующее недифференцируемые метрики качества (такие как GLEU).

2. Учет лексики семантики (например, предложение "По реке плывет кирпич." семантически бессмысленно).

3. Учет контекста за пределами рассматриваемого предложения для исключения неоднозначности ответа.

Исследованные в ходе работы технологии планируется применять в мобильном приложении по обучению языкам на основе технологий обработки естественного языка.

# Библиография


[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. sep 2014.

[2] Timothy Baldwin, Paul Cook, Marco Lui, Andrew Mackinlay, and Li Wang. How Noisy Social Media Text, How Diffrnt Social Media Sources ? *Proc. IJCNLP 2013*, (October):356–364, 2013.

[3] Satanjeev Banerjee and Alon Lavie. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 0(June):65–72, 2005.

[4] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[5] Jeroen Blaak and Beverley Jackson. Literacy in Everyday Life. pages 1–441, 2010.

[6] Thorsten Brants, Ashok C Popat, Peng Xu, Franz J Och, and Jeffrey Dean. Large Language Models in Machine Translation. *Google, Inc.*, 1(June):858–867, 2007.

[7] C Bryant and H T Ng. How far are we from fully automatic high quality grammatical error correction? *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, 1:697–707, 2015.

[8] William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, Attend, Spell. aug 2015.

[9] Qiming Chen and Ren Wu. CNN Is All You Need. dec 2017.

[10] Noam Chomsky. *Aspects of the Theory of Syntax*, volume 11. 2014.

[11] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. dec 2014.

[12] Daniel Dahlmeier, Ht Ng, and Sm Wu. Building a large annotated corpus of learner English: The NUS corpus of learner English. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, 2013.

[13] Rumen Dangovski, Li Jing, and Marin Soljacic. Rotational Unit of Memory. pages 1–14, 2017.

[14] Hal Daumé and Daniel Marcu. Learning as Search Optimization: Approximate Large Margin Methods for Structured Prediction. jul 2009.

[15] B Flanagan, C Yin, T Suzuki, and S Hirokawa. Classification and Clustering English Writing Errors Based on Native Language. In *2014 IIAI 3rd International Conference on Advanced Applied Informatics*, pages 318–323, 2014.

[16] Qin Gao and Stephan Vogel. Parallel Implementations of Word Alignment Tool. *SETQA-NLP '08 Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, (June):49–57, 2008.

[17] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Pmlr*, 9:249–256, 2010.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[19] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutn\'\ik, Bas R Steunebrink, and Jürgen Schmidhuber. {LSTM:} A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems. (Preprint arXiv:1503.04069, 2015)*, pages 1–11, 2016.

[20] Kenneth Heafield. KenLM : Faster and Smaller Language Model Queries. *Proceedings of the Sixth Workshop on Statistical Machine Translation*, (2009):187–197, 2011.

[21] Sepp Hochreiter and J Urgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[22] Matthew Honnibal and Mark Johnson. An Improved Non-monotonic Transition System for Dependency Parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal, 2015. Association for Computational Linguistics.

[23] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 2017.

[24] Marcin Junczys-Dowmunt and Roman Grundkiewicz. Phrase-based Machine Translation is State-of-the-Art for Automatic Grammatical Error Correction. may 2016.

[25] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task. apr 2018.

[26] Masahiro Kaneko, Yuya Sakaizawa, and Mamoru Komachi. Grammatical Error Detection Using Error- and Grammaticality-Specific Word Embeddings. *Association for Computational Linguistics*, (2016):40–48, 2017.

[27] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and Understanding Recurrent Networks. jun 2015.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. dec 2014.

[29] Saiprasad Koturwar and Shabbir Merchant. Weight Initialization of Deep Neural Networks(DNNs) using Data Statistics. oct 2017.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks, 2012.

[31] Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor Forcing: A New Algorithm for Training Recurrent Networks. oct 2016.

[32] Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. SEARNN: Training RNNs with Global-Local Losses. jun 2017.

[33] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully Character-Level Neural Machine Translation without Explicit Segmentation. oct 2016.

[34] Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. Character-based Neural Machine Translation. nov 2015.

[35] Adam Lopez. Statistical machine translation. *ACM Computing Surveys*, 40(3):1–49, 2008.

[36] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. aug 2015.

[37] Minh-Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. Addressing the Rare Word Problem in Neural Machine Translation. oct 2014.

[38] Mitchell P Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[39] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. jan 2013.

[40] Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction. *Transactions of the Japanese Society for Artificial Intelligence*, 28(5):420–432, 2013.

[41] Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. GLEU: Automatic Evaluation of Sentence-Level Fluency. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, 2007.

[42] Daniel Naber. A Rule-Based Style and Grammar Checker. pages 1–77, 2003.

[43] D. Nadeau. A survey of named entity recognition and classification. *Linguisticae Investigationes*, (30):3–26., 2007.

[44] Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. *Proceedings of the 27th International Conference on Machine Learning*, (3):807–814, 2010.

[45] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. GLEU Without Tuning. may 2016.

[46] Hwee Tou Ng, Yuanbin Wu, and Christian Hadiwinoto. The CoNLL-2013 Shared Task on Grammatical Error Correction. *Computational Natural Language Learning (CoNLL), Shared Task*, pages 1–12, 2013.

[47] Kishore Papineni, Salim Roukos, Todd Ward, and Wj Zhu. BLEU: a method for automatic evaluation of machine translation. *... of the 40Th Annual Meeting on ...*, (July):311–318, 2002.

[48] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training Recurrent Neural Networks. nov 2012.

[49] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017.

[50] Ofir Press and Lior Wolf. Using the Output Embedding to Improve Language Models. aug 2016.

[51] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. jun 2016.

[52] Alla Rozovskaya and Dan Roth. Building a State-of-the-Art Grammatical Error Correction System. *Transactions of the Association for Computational Linguistics (TACL)*, 2:419–434, 2014.

[53] Sebastian Ruder and Barbara Plank. Learning to select data for transfer learning with Bayesian Optimization. 2017.

[54] Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. Grammatical Error Correction with Neural Reinforcement Learning. 2017.

[55] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving Neural Machine Translation Models with Monolingual Data. nov 2015.

[56] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. aug 2015.

[57] Richard S Sutton and Andrew G Barto. Reinforcement Learning : An Introduction. 2015.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. 2017.

[59] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. sep 2016.

[60] Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y. Ng. Neural Language Correction with Character-Based Attention. mar 2016.

[61] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A New Dataset and Method for Automatically Grading ESOL Texts. *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, 2011.

[62] Li Yujian and Liu Bo. A Normalized Levenshtein Distance Metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1091–1095, jun 2007.

[63] Barret Zoph and Quoc V. Le. Neural Architecture Search with Reinforcement Learning. nov 2016.

[64] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer Learning for Low-Resource Neural Machine Translation. apr 2016.