**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ» (ИУ)

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ» (ИУ7)

# РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## *К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ*

## *НА ТЕМУ:*

## *Методы машинного перевода европейских языков*

Студент **ИУ7-53Б** _____ **Кочуйков И. И.**

Студент **ИУ7-53Б** _____ **Лысцев Н. Д.**

Студент **ИУ7-53Б** _____ **Лазутин А. В.**

Студент **ИУ7-53Б** _____ **Князев Д. Ю.**

Руководитель _____ **Кострицкий А. С.**

*2023 г.*

# СОДЕРЖАНИЕ

# ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В текущей расчетно-пояснительной записке применяется следующие сокращения и обозначения:

МП — машинный перевод;

СМП — статистический машинный перевод;

НМП — нейронный машинный перевод;

RNN — Recurrent Neural Network;

LSTM — Long short-term memory;

FFN — Feedforward Neural Networ;

BLEU — Bilingual Evaluation Understudy.

# ВВЕДЕНИЕ

Естественный язык — это язык, используемый для общения людей, в отличие от формальных языков и других типов знаковых систем, и не созданный целенаправленно, в отличие от искусственных языков [1]. Под европейскими языками подразумеваются те естественные языки, которые являются официальными в деятельности Европейского союза [2]. Основные признаки европейских языков:

– присутствует письменность;

– предложения читаются и записываются слева-направо;

– алфавит состоит из знаков, не имеющих лексического смысла, называемых буквами.

В современном мире существует большое количество статей, технической документации, книг, написанных на разных европейских языках. Перевод таких текстов с одного языка на другой человеком вручную требует значительных временных и финансовых затрат, а также наличия соответствующих навыков. Для уменьшения этих затрат можно автоматизировать процесс перевода [3].

Одним из способов автоматизации является использование систем машинного перевода, в которых перевод некоторого текста с одного естественного языка на другой, реализуется компьютером полностью или почти полностью. В процессе машинного перевода на вход машины подается текст, словесная часть которого не сопровождается никакими дополнительными указаниями, а на выходе получается текст на другом языке, являющийся переводом входного, причем преобразование входного текста в выходной происходит без вмешательства человека [4].

Целью работы является проведение анализа существующих методов машинного перевода европейских языков.

Для достижения поставленной цели необходимо решить следующие задачи:

– провести анализ предметной области машинного перевода европейских языков;

– описать основные подходы решения задачи машинного перевода европейских языков;

– сформулировать критерии сравнения применяемых методов и выполнить их классификацию.

# 1 Системы машинного перевода на основе правил

Машинный перевод на основе правил — один из методов машинного перевода, который основан на использовании правил грамматики языков [5]. В процессе перевода осуществляется анализ, согласно правилам исходного языка, и синтез, согласно правилам языка перевода. Существует три подхода к машинному переводу на основе правил: системы пословного перевода, трансферные системы и интерлингвистические системы [6].

## 1.1 Системы пословного перевода

При использовании систем пословного перевода предполагается, что основные различия между языками являются лексическими [5]. Синтаксического анализа при переводе не проводится, правила перестановки слов могут определяться словарем или быть заданы в коде программы [6]. Данный подход можно описать следующей последовательностью действий [5]:

1) согласно переводному словарю, слова предложения из исходного языка переводятся в язык перевода;

2) порядок слов изменяется, согласно правилам языка перевода.

Данный подход является наиболее простым, так как не учитывает морфологические различия языков, но имеет недостатки в виде ограничения на языки, согласно указанному ранее предположению, а также невысокой сложности языков, к которым применим подход, из-за отсутствия полноценного синтаксического анализа [5].

## 1.2 Трансферные системы

Трансферный подход позволяет получить более точный перевод в сравнении с системами пословного перевода без наложения ограничений на рассматриваемые языки благодаря синтаксическому анализу [6]. Для перевода используется некоторое промежуточное представление текста, одним из которых может выступать синтаксическое дерево — структура, в которой предложение разбивается на компоненты — непосредственные составляющие [7; 8]. Пример синтаксического дерева для предложения на английском языке можно увидеть на 1.1.

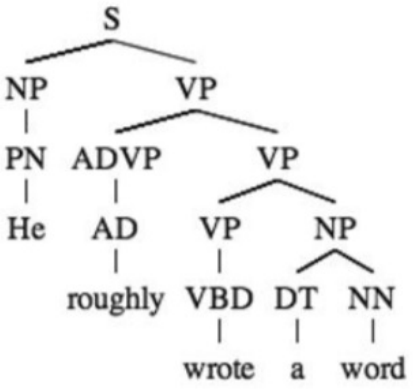


Рисунок 1.1 – Синтаксическое дерево [5]

В данном примере предложение «He roughly wrote a word» разделяется на именную группу (NP) «He» и глагольную группу (VP) «roughly wrote a word». В свою очередь именная группа состоит из местоимения (PN) «He», а глагольная группа – на наречную группу (ADVP), «roughly», являющуюся одним наречием (AD), и другую глагольную группу «wrote a word», которая в свою очередь раскладывается на глагольную группу, состоящую из глагола прошедшего времени (VBD) «wrote», и именную группу «a word», состоящую из артикля «a» (DT) и существительного «word» [5]. Трансферные системы работают следующим образом [7]:

1) Из предложения на основе правил исходного языка строится синтаксическое дерево;

2) Согласно правилам перевода из исходного языка в язык перевода, исходное синтаксическое дерево преобразуется в синтаксическое дерево языка перевода;

3) По синтаксическому дереву строится предложение на языке перевода.

## 1.3 Интерлингвистические системы

Интерлингвистические системы при переводе используют интерлингву – некоторое промежуточное представление текста, независимое от грамматик исходного языка и языка перевода [6]. Работу интерлингвистической системы можно описать следующим образом [7]:

1) Путем анализа на основе правил исходного языка смысловое содержание предложения представляется в виде текста интерлингвы;

2) Согласно правилам языка перевода из текста интерлингвы синтезируется предложение на языке перевода.

Основной проблемой данного подхода является создание интерлингвы [9], из-за чего в некоторых системах в качестве интерлингвы используются естественный язык, например английский, однако такой подход приводит к неточностям, так как и при переводе в промежуточный язык из исходного и переводе в язык перевода из промежуточного может теряться смысл [10].

# 2 Статистический машинный перевод. Методы статистического машинного перевода

## 2.1 Статистический машинный перевод

Статистический машинный перевод — разновидность МП, которая основана на поиске наиболее вероятного перевода предложения с использованием данных, полученных из двуязычной совокупности текстов (параллельного корпуса) в результате обучения (по языковым парам) [3].

При статистическом подходе используется такое понятие как «канал помехами»(Noisy-Channel Model) [11]: рассматривая перевод с английского на русский, предполагается, что английское предложение на самом деле – русское предложение, но искаженное неким шумом. Для корректного перевода или же «расшифровки шума» необходимо знать, что в рассматриваемом случае говорят русскоязычные и как рассматриваемый текст искажается, что превращается в текст на английском языке. Перевод осуществляется после поиска такого русского предложения, которое максимизирует вероятность «встречи» английского предложения и соответствующего ему русского предложения. В основе такого поиска лежит теорема Байеса:

$$argmax_e P(e|f) = argmax_e P(f|e) \cdot P(e), \text{ где} \tag{2.1}$$

$e$ — *предложение перевода,* $f$ — *предложение оригинала.*

Таким образом, в основе перевода, то есть вероятности того, что предложение оригинала $f$ будет переведено конечным предложением $e$, лежат модель языка ($P(e)$ в формуле 2.6), и модель перевода — ($P(f|e)$ в формуле 2.6).

Модель языка должна присваивать оценку вероятности любому предложению конечного языка, а модель перевода должна присваивать оценку вероятности предложения оригинала при условии определенного предложения на конечном языке [3].

Таким образом, задача, которая ставится перед системой статистического машинного перевода — не перевод, как таковой, а декодирование. То есть, в данном случае, перевод — не что иное, как расшифровка исходного текста.

Формально, алгоритм работы системы статистического машинного перевода можно описать в следующих шагах:

1) Составляются параллельные корпусы [3] для языка источника и языка перевода;

2) Производится выравнивание [3] параллельных корпусов текстов;

3) Согласно выбранной модели перевода ищутся такие значения таблиц переводных соответствий [11], которые максимизируют вероятность части корпуса источника при имеющейся части корпуса языка перевода;

4) На основе корпуса языка перевода составляется модель языка;

5) На основе полученных данных для незнакомого предложения на языке источника ищется предложение на языке перевода, максимизирующее произведение вероятностей, присваеваемых моделью языка и моделью перевода.

Алгоритм перевода можно изобразить следующей схемой.


Рисунок 2.1 – Алгоритм работы системы статистического машинного перевода

Итак, весь алгоритм основывается на выравнивании, создании моделей и самом переводе. Рассмотрим данные составляющие более подробно.

### 2.1.1 Выравнивание параллельных корпусов

Параллельный текст — текст на одном языке вместе с его переводом на другой язык. Параллельный корпус — большие собрания параллельных текстов [3].

Выравнивание параллельных корпусов — процесс, в котором для каждого параллельного текста каждому его предложению на языке источника ставится в соответствие предложение на языке перевода.

Для выравнивания текстов система использует инструментальное средство автоматического выравнивания параллельных текстов (alignment tool). С помощью таких инструментов система может проанализировать и привести в соответствие тексты оригинала и перевода [3].

Подготовленные таким образом тексты используются для обучения модели перевода.

### 2.1.2 Модель перевода

Модель перевода, или таблица перевода — это таблица-словарь, в которой для всех известных системе слов и фраз на одном языке перечислены все возможные их переводы на другой язык и указана вероятность этих переводов [11].

### 2.1.3 Модель языка

Модель языка оценивает фразы целевого языка и дает им соответствующую вероятность. В качестве модели языка в системах статистического машинного перевода используются преимущественно используются различные модификации $n$-граммной модели, утверждающей, что грамматичность выбора очередного слова определяется лишь тем, какие $(n-1)$ слов идут перед ним. Вероятность каждого $n$-грамма определяется по его встречаемости в параллельном корпусе [3].

Рассмотрим математику $n$-граммных моделей с $n = 3$:

$$\begin{gathered} P(e) = P(e_1, e_2, \ldots, e_n) = \\ P(e_1) \cdot P(e_2|e_1) \cdots P(e_n|e_1, e_2, \ldots, e_{n-1}) \simeq \\ P(e_1) \cdot P(e_2|e_1) \cdots P(e_n|e_{n-2}, e_{n-1}), \text{ где} \end{gathered} \tag{2.2}$$

*$e$ — предложение на языке перевода, $P(e)$ — вероятность перевода всего предложения $e$, $e_i$ — слово в предложении $e$, $i = \overline{1,n}$, $P(e_i)$ — вероятность перевода слова $e_i$, $i = \overline{1,n}$.*

### 2.1.4 Декодер

Декодер — составляющая переводчика, которая непосредственно переводом. Для каждого предложения исходного текста он подбирает все варианты перевода, сочетая между собой фразы из модели перевода, и сортирует их по убыванию вероятности. Затем все получившиеся варианты декодер оценивает с помощью модели языка.

## 2.2 Методы статистического машинного перевода

В статистическом машинном переводе существует два основных метода перевода:

– метод перевода, основанный на словах;

– метод перевода, основанный на фразах.

Каждому из вышеперечисленных методов соответствуют свои модели перевода. Рассмотрим каждый из методов более подробно.

### 2.2.1 Метод перевода, основанный на словах

Введем некоторые обозначения:

– $f$ — слово на языке источника, с которого нужно переводить;

– $e$ — слово на языке приемника, на который нужно переводить.

Понятие статистического машинного перевода подразумевает использование статистики. На основе анализа параллельных корпусов для каждого слова $f$ в предложении выполняется подсчет количества переводов для каждого возможного варианта перевода $e$ на язык приемника. Далее, на основе этих подсчетов, производится оценка распределения вероятности лексического перевода.

Формально, находится функция [11]

$$p_f : e \to p_f(e), \tag{2.3}$$

что для слова $f$ возвращает вероятность для каждого выбора перевода $e$, которая указывает, насколько вероятен этот перевод.

Функция 2.3 должна обладать следующим свойством:

$$\sum_e p_f(e) = 1. \tag{2.4}$$

После получения вероятностного распределения для каждого слова $f$ в предложении выбирается наиболее вероятный перевод $e$. Сопоставление

слова $f$ и его перевода $e$ в предложении на языках источника и приемника определяются функцией выравнивания:

$$a : j \to i, \tag{2.5}$$

которая отображает каждое выходное слово в позиции $i$ к входному слову в позиции $j$.

## IBM Model 1

Генеративный метод моделирования — метод разбиение процесса генерации данных на более мелкие шаги, моделирование более мелких шагов с помощью вероятностных распределений и объединение шагов в последовательную историю [11].

Поскольку прямое моделирование распределения вероятностей перевода для полных предложений сложно оценить (большинство предложений встречается только один раз, даже в больших текстовых коллекциях), применяется генеративный метод моделирования: процесс разбивается на более мелкие этапы, в нашем случае на перевод отдельных слов, а их расстановку в правильном порядке обеспечит модель языка.

IBM Model 1 — генеративная модель перевода предложений, основанная исключительно на распределении вероятностей лексического перевода [11]. Данная модель генерирует несколько различных переводов предложения, каждый из которых имеет разную вероятность. Единственным массивом данных, которым оперирует IBM Model 1 является таблица вероятностей попарно переводимых соответствий слов двух языков [3].

Обучение IBM Model 1 производится на параллельных корпусах, выровненных н уровне предложений. IBM Model 1 допускает ситуацию, в которой наиболее употребительным переводом нескольких смысловых слов может быть признано одно высокочастотное — например, служебное — слово конечного языка, и данная модель не всегда правильно учитывает порядок слов в предложении [3].

### IBM Model 2

Чтобы сохранить при переводе информацию, заключенную в порядке слов, была предложена IBM Model 2.

В IBM Model 2 помимо таблица вероятностей попарно переводимых соответствий слов двух языков вводится таблица распределения вероятностей выравнивания, то есть вероятностей, что при определенной длине предложения на языке перевода $l_e$ и длине предложения на языке источника $l_f$ отображает каждое выходное слово в позиции $i$ к входному слову в позиции $j$.

Таким образом, перевод с помощью IBM Model 2 можно описать в двух этапах:

1) Выполнение лексического перевода, как в IBM Model 1;

2) Выполнение этапа выравнивания.

## 2.2.2 Метод перевода, основанный на фразах

Наиболее эффективные в настоящее время системы статистического машинного перевода основаны на моделях, основанных на фразах: моделях, которые переводят небольшие последовательности слов за раз [11].

Под фразой понимается любая многословная единица предложения [11]. В методе перевода, основанном на фразах, наименьшей единицей перевода является не слово, как в первом методе, а фраза.

Алгоритм перевода следующий:

1) Входное предложение на языке источника разбивается на фразы;

2) Каждая фраза переводится на фразу в языке перевода;

3) Фразы на языке перевода переупорядочиваются согласно с моделью языка.

В основе перевода на основе фраз лежит правило Байеса [11]:

$$e_{best} = argmax_e P(e|f) = argmax_e P(f|e) \cdot P_{LM}(e), \text{ где} \tag{2.6}$$

$e_{best}$ — *наиболее вероятный перевод предложения* $f$ *на языке источника,* $P_{LM}(e)$ — *модель языка,* $P(f|e)$ — *модель перевода.*

Данный метод имеет несколько преимуществ. Во-первых, слова, возможно, не являются лучшим атомарным средством единиц для перевода из-за частых сопоставлений один ко многим (и наоборот).Во-вторых, перевод групп слов вместо отдельных слов помогает устранить неоднозначность перевода. Есть и третье преимущество: если у нас есть большие учебные корпуса, мы можем выучить все более и более длинные полезные фразы, иногда даже запоминать перевод целых предложений [11].

### IBM Model 3

IBM Model 2 не допускает возможности, что одному слову из предложения на языке источника соответствует соответствует несколько слов в предложении на языке перевода. Этот недостаток устраняется в IBM Model 3, где вводится понятие коэффициента деления слова оригинала, и, соответственно, таблица вероятностей каждого значения коэффициента деления для каждого слова [3].

### IBM Model 4 и IBM Model 5

В IBM Model 4 и близкой к ней IBM Model 5 делается следующий шаг к включению понятия грамматики в систему статистического машинного перевода. В IBM Model 4 появляется понятие класса слов, определяемое автоматически для всех слов языка оригинала и языка перевода.

# 3 Рекуррентные нейронные сети и сети долгой краткосрочной памяти, используемые для машинного перевода

Рекуррентная нейронная сеть (Recurrent Neural Network, RNN) — популярный вид нейронных сетей, используемых в обработке естественного языка. Рекуррентная нейронная сеть оценивает произвольные предложения на основе того, насколько часто они встречались в текстах. Это дает меру грамматической и семантической корректности, что позволяет использовать такие модели для перевода текстов. Кроме того, такие модели генерируют новый текст [12].

## 3.1 Рекуррентные нейронные сети

Нейронная сеть прямой связи — характеризуется направлением потока информации между ее слоями. Слой — это набор нейронов, которые выполняют определенную задачу [13]. Нейрон — вычислительная единица, получающая информацию и производящая над этой информацией какие-либо вычисления [13; 14]. Поток нейронной сети прямой связи однонаправленный, что означает, что информация в модели течет только в одном направлении — вперед — от входных узлов, через скрытые узлы (если таковые имеются) и к выходным узлам, без каких-либо циклов или петель, в отличие от рекуррентны (рис. 3.1). Скрытые слои нейронной сети — это слои, которые не являются ни входными, ни выходными, а являются промежуточными шагами в вычислениях сети [15]. Нейронные сети с прямой связью не позволяют получать последовательности переменной длины в качестве входных данных, что ограничивает возможности модели. Поскольку RNN допускают последовательности переменной длины как на входе, так и на выходе, они являются следующим шагом в статистическом моделировании языка [16].

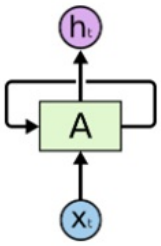


Рисунок 3.1 – Цикл в рекуррентной нейронной сети

На приведенном выше рисунке 3.1 фрагмент нейронной сети, $A$, рассматривает некоторые входные данные $x_t$ и выводит значение $h_t$. Цикл позволяет передавать информацию с одного этапа сети на следующий.

## 3.2 Проблема долгосрочных зависимостей

Иногда нам нужно только просмотреть свежую информацию, чтобы выполнить текущую задачу. Например, рассмотрим языковую модель, пытающуюся предсказать следующее слово на основе предыдущих. Если мы пытаемся предсказать последнее слово в предложении «the clouds are in the sky», нам не нужен какой–либо дополнительный контекст — следующим словом будет «sky». В таких случаях, когда разрыв между релевантной информацией и местом, где она необходима, невелик, RNN могут научиться использовать предыдущую информацию.

Но есть и случаи, когда нам нужно больше информации о контексте. Попробуйте предсказать последнее слово в тексте «Я вырос во Франции... Я свободно говорю по-французски». Недавняя информация предполагает, что следующее слово, вероятно, является названием языка, но если мы хотим сузить круг языков, нам нужен контекст «Франция», начиная с более ранних времен. Вполне возможно, что разрыв между релевантной информацией и моментом, когда она необходима, станет очень большим [16]. К сожалению, по мере роста этого разрыва RNN становятся неспособными научиться соединять информацию (рис. 3.2).

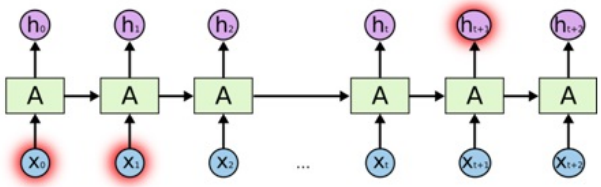


Рисунок 3.2 – Проблема долгосрочных зависимостей

## 3.3 Сети долгой краткосрочной памяти

Сети с долговременной кратковременной памятью (Long short-term memory, LSTM), обычно называемые LSTM, представляют собой особый вид RNN, способный изучать долгосрочные зависимости [17].

Все рекуррентные нейронные сети имеют форму цепочки повторяющихся фрагментов нейронной сети. В стандартных RNN этот повторяющийся модуль будет иметь структуру, такую как один слой tanh (рис. 3.3).

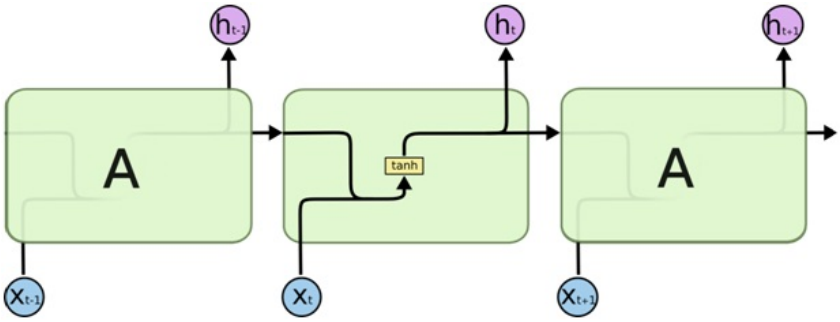


Рисунок 3.3 – Повторяющийся модуль в RNN, содержащий только один уровень

LSTM также имеют структуру, подобную цепочке на рис. 3.4, но повторяющийся модуль имеет другую структуру. Вместо одного слоя нейронной сети их четыре [16].

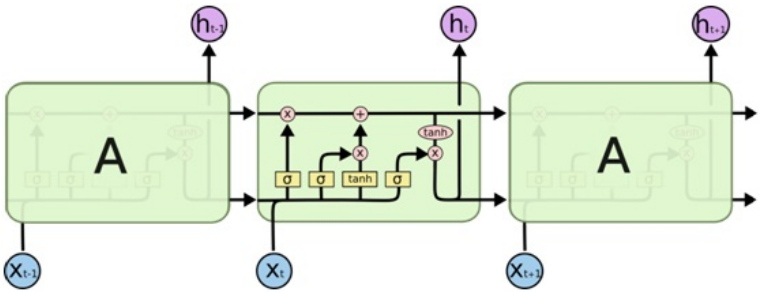


Рисунок 3.4 – Повторяющийся модуль в LSTM, содержащий четыре взаимодействующих уровня

Любой блок нейронной сети — это нейрон. Он работает по следующему принципу:

1) Нейрон получает входные значения $x_i$ , которые могут выглядеть как векторные или числовые значения, представляющие признаки или значения от предыдущих нейронов.

2) Входное значение умножается на соответствующий вес. Этот вес $(W_{ij})$ определяет важность этого входа для вычислений нейрона. Полученные в результате умножения значения суммируются:

$$\sum_{i=1}^{n} \omega_{ji} x_i. \tag{3.1}$$

3) К сумме взвешенных значений добавляется смещение. Смещение $(b_i)$ в нейроне — это дополнительный параметр, некоторое константное значение, благодаря которому нейрон может активироваться, даже если взвешенные входы принимают нулевое значение.

4) Полученная сумма передается через функцию активации, которая применяет нелинейное преобразование к взвешенной сумме. Функция активации определяет выходной сигнал нейрона, добавляя нелинейности в модель.

В сетях долгой краткосрочной памяти используются две функции активации [18]:

– сигмоидная функция (логистическая). Нелинейна логистическая функция, которая сжимает входные значения от 0 до 1 по формуле:

$$F(x) = \frac{1}{1 + exp(-x)}, \text{ где} \tag{3.2}$$

*x — входное значение;*

– гиперболический тангенс (Tahn). Нелинейна логистическая функция, которая сжимает входные значения от -1 до 1 по формуле:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \text{ где} \tag{3.3}$$

*x — входное значение.*

## 3.4 Основная идея, лежащая в основе LSTM

Ключом к LSTM является состояние ячейки, линия, проходящая через верхнюю часть диаграммы (рис. 3.5). Благодаря ее наличию LSTM-блок имеет возможность сохранять и передавать долгосрочный контекст дальше по рекурсии [19]. Она проходит прямо по всей цепочке, лишь с некоторыми незначительными линейными взаимодействиями [16].

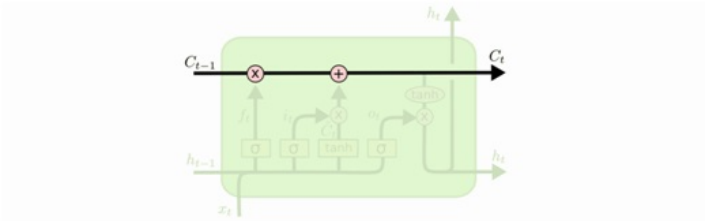

Рисунок 3.5 – «Конвейерная лента»

LSTM действительно обладает способностью удалять или добавлять информацию о состоянии ячейки, тщательно регулируемую структурами, называемыми гейтами (рис. 3.6).

Гейты — это способ необязательного пропускания информации. Они состоят из слоя сигмовидной нейронной сети и операции поточечного умножения [13; 16].

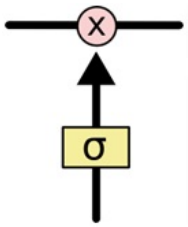


Рисунок 3.6 – Гейт

Уровень sigmoid выводит числа от нуля до единицы, описывающие, сколько каждого компонента должно быть пропущено. Значение ноль означает «ничего не пропускать», а значение единица означает «пропускать все»

LSTM имеет три таких уровня для защиты и контроля состояния ячейки [16].

## 3.5 Пошаговое прохождение через LSTM

### 3.5.1 Нахождение информации для освобождения

Первый шаг в LSTM (рис. 3.7) — решить, какую информацию мы собираемся выбросить из состояния ячейки. Это решение принимается сигмоидным слоем. Он получает $h_{t-1}$ и $x_t$ и выводит число в диапазоне от 0 до 1 для каждого номера в состоянии ячейки $C_{t-1}$. 1 означает «полностью сохранить это», а 0 означает «полностью избавиться от этого».

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \text{ где} \tag{3.4}$$

*$f_t$ — вектор слоя забывания, вес запоминания старой информации, $W_f$ — весовая матрица нейронной сети, $h_t$ — скрытый вектор состояния, также известный как выходной вектор, $x_t$ — вектор входных значений, $b_f$ — вектор смещения нейронной сети.*

Давайте вернемся к нашему примеру языковой модели, пытающейся предсказать следующее слово на основе всех предыдущих. В такой задаче состояние ячейки может включать в себя пол текущего субъекта, чтобы можно было использовать правильные местоимения. Когда мы видим нового субъекта, мы хотим забыть пол старого субъекта [16].

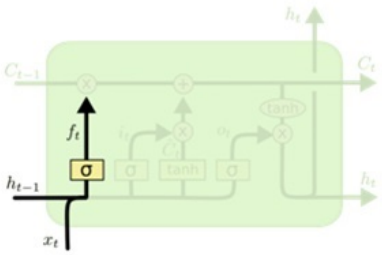


Рисунок 3.7 – Слой забывания

### 3.5.2 Нахождение новой информации для добавления

Следующий шаг (рис. 3.8) — решить, какую новую информацию мы собираемся сохранить в состоянии ячейки. Это состоит из двух частей. Сигмоидный слой решает, какие значения мы будем обновлять. Затем слой $tanh$ создает вектор новых значений-кандидатов, $\tilde{C}_t$, которые надо добавить в состояние. На следующем шаге мы объединим эти два, чтобы создать обновление состояния.

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \end{aligned}, \textit{где} \tag{3.5}$$

*$i_t$ — вектор входного слоя, вес получения новой информации, $W_i, W_C$ — весовый матрицы нейронной сети, $b_i, b_C$ — вектора смещения нейронной сети, $C_t$ — вектор состояния ячейки, $\tilde{C}_t$ — вектор нового состояния ячейки.*

В примере нашей языковой модели мы хотели бы добавить пол нового субъекта в состояние ячейки, чтобы заменить старый, который мы забываем [16].

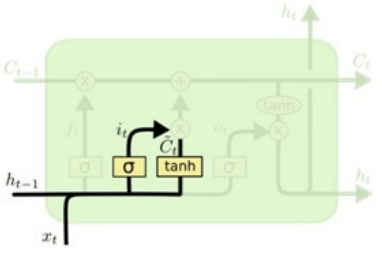


Рисунок 3.8 – Создание обновленного состояния

### 3.5.3 Обновление старого состояния ячейки

Теперь пришло время обновить старое состояние ячейки (рис. 3.9), $C_{t-1}$ в новое состояние ячейки $C_t$. Мы умножаем старое состояние на $f_t$, забываем то, что мы решили забыть ранее. Затем мы добавляем $i_t \cdot \tilde{C}_t$. Это новые значения-кандидаты, масштабируемые в зависимости от того, насколько мы решили обновить значение каждого состояния.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t. \tag{3.6}$$

В случае языковой модели именно здесь мы фактически удаляем информацию о поле старого субъекта и добавляем новую информацию, как мы решили на предыдущих шагах [16].

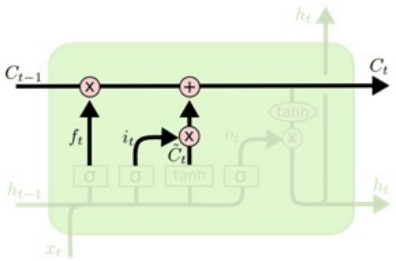


Рисунок 3.9 – Обновление старого состояние

### 3.5.4 Нахождение информации для вывода

Наконец, нам нужно решить, что мы будем выводить (рис. 3.10). Этот вывод будет основан на состоянии нашей ячейки, но будет отфильтрованной. Сначала мы запускаем сигмовидный слой, который решает, какие части состояния ячейки мы собираемся выводить. Затем мы вводим состояние ячейки через слой $tanh$ и умножаем это на выходные данные сигмовидного слоя, чтобы мы выводили только те части, которые решили использовать.

$$\begin{gathered} o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t = o_t \cdot tanh(C_t) \end{gathered}, \textit{где} \tag{3.7}$$

*$o_t$ — вектор выходного слоя, $W_o$ — весовая матрица нейронной сети, $b_o$ — вектор смещения нейронной сети.*

Для примера языковой модели, поскольку она только что увидела тему, она может захотеть вывести информацию, относящуюся к глаголу, на случай, если это то, что будет дальше. Например, он может выводить, является ли подлежащее единственным или множественным, чтобы мы знали, в какую форму следует спрягать глагол, если это то, что следует дальше [16].

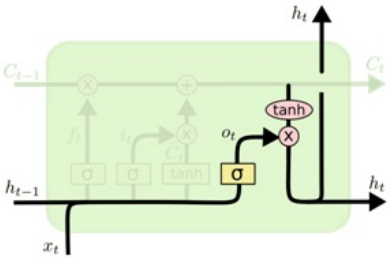


Рисунок 3.10 – Вывод отфильтрованного состояния ячейки

# 4 Архитектура нейронных сетей Трансформер

Качество переведенного текста для большинства языковых пар удалось повысить благодаря использованию математических моделей, схожих по набору составляющих элементов, принципу функционирования и организации с биологическими нейронными сетями, такие модели называют искусственными нейронными сетями или нейронными сетями [20—24]. Перевод с использованием данных моделей называется нейронным машинным переводом. Одним из современных подходов к НМП является использование архитектуры Трансформер [25]. Модель, основанную на архитектуре Трансформер, как и любую другую модель, перед началом использования необходимо обучить. Одним из результатов обучения является набор слов, участвовавших в качестве обучающего материала. Эти наборы слов, называемые словарями, используются для перевода предложения с естественного языка во внутреннее представление модели и наоборот [26].

## 4.1 Архитектура Трансформера

Архитектура Трансформер включает в себя следующие компоненты [27]:

– механизм, преобразующий текст на исходном языке во внутреннее представление модели, называемый кодирующим компонентом;

– механизм, преобразующий текст из внутреннего представления модели в текст на целевом языке, называемый декодирующим компонентом.

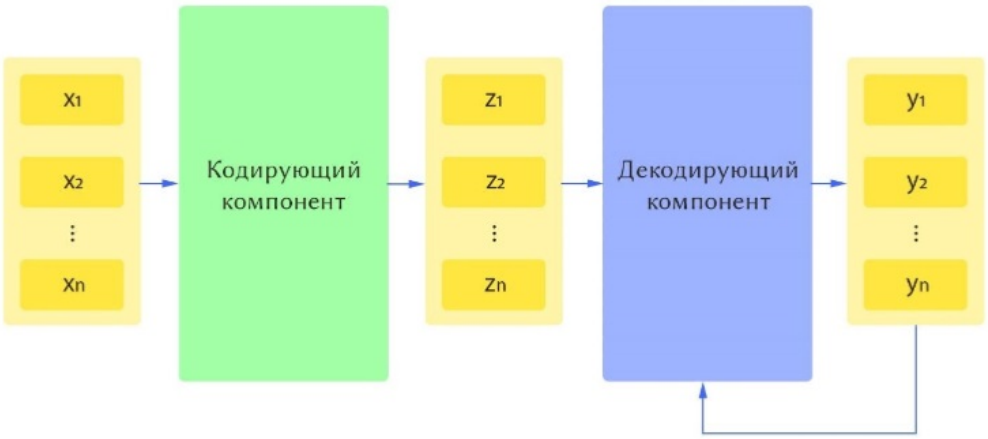


Рисунок 4.1 – Изображение системы, состоящей из кодирующего и декодирующего компонентов

На 4.1 входными данными для кодирующего компонента обозначена последовательность элементов $x = (x_0, x_1, \ldots, x_n)$, содержащая информацию об исходном тексте. Кодирующий компонент переводит эти данные в последовательность $z = (z_0, z_1, \ldots, z_n)$, являющуюся внутренним представлением модели. Декодирующий компонент, получив на вход $z$, генерирует выходную последовательность $y = (y_0, y_1, \ldots, y_n)$ по одному элементу за раз. При генерации нового элемента декодирующий компонент учитывает уже полученную им на предыдущих шагах выходную последовательность.

## 4.2 Подготовка входных данных

Перед началом работы декодирующего компонента текст на исходном языке необходимо подготовить. В процессе подготовки текст на исходном языке разделяется на последовательность предложений, а предложения на последовательности слов. Каждое слово преобразуется в число, соответствующее позиции в словаре модели. Если слово не встречается в словаре, то оно разбивается на две части и более на основе лингвистических правил исходного языка с расчетом на то, что полученные подвыражения будут находиться в словаре. Например, от слова отделяется суффикс, предлог или окончание. Дополнительно, в качестве первого и последнего элементов, в данную последовательность добавляются специальные символы, не встречающиеся в словарях исходного и целевого языков, цель которых заключается в явном определении

начала и конца переводимого предложения. Набор элементов, полученный на данном этапе, называется набором токенов, а сам процесс — токенизацией [28]. Каждый токен, будучи скалярной величиной, преобразуется в одномерный массив чисел, называемый вложением или вектором, данный этап называется встраиванием [26].

В предложениях на европейских языках важен порядок слов. Чтобы сохранить эту информацию во вложениях, поступающих на вход кодирующему компоненту, используется способ, называемый позиционным кодированием. Согласно алгоритму, вычисляется вектор, связанный с позицией исходного вложения, размерности, обозначаемой $d_{model}$, такой же, как и у самих вложений, обеспечивая тем самым возможность их дальнейшего сложения [25].

Одно из возможных решений предполагает сложение элементов векторов со значениями следующих функций [25]:

$$\begin{aligned} PE_{(position,2i)} &= \sin[(\frac{position}{10000})^{\frac{2i}{d_{model}}}] \\ PE_{(position,2i+1)} &= \cos[(\frac{position}{10000})^{\frac{2i}{d_{model}}}] \end{aligned}, \textit{где} \tag{4.1}$$

*position — порядковый номера вложения в предложении, i — индекса элемента вложения.*

В результате применения к вложению формул 4.1 будет получен вектор, в котором сохранена информация об исходном токене и его позиции в предложении.

Таким образом, алгоритм получения входной последовательности $(x_0, x_1, \ldots, x_n)$ кодирующего компонента описывается следующими шагами [27]:

1) текст на исходном языке разбивается на предложения;

2) каждое предложение разбивается на слова. Ко всем словам предложения применяются следующие действия:

   2.1) из слова с помощью токенизации формируется токен;

   2.2) из токена с помощью встраивания генерируется вложение;

   2.3) к полученному вложению применяется позиционное кодирование.

В результате работы данного алгоритма генерируется последовательность векторов, готовая для передачи на вход кодирующему компоненту.

## 4.3 Кодирующий компонент

Как кодирующий, так и декодирующий компоненты состоят из подсистем, называемых наборами кодировщиков и декодировщиков.

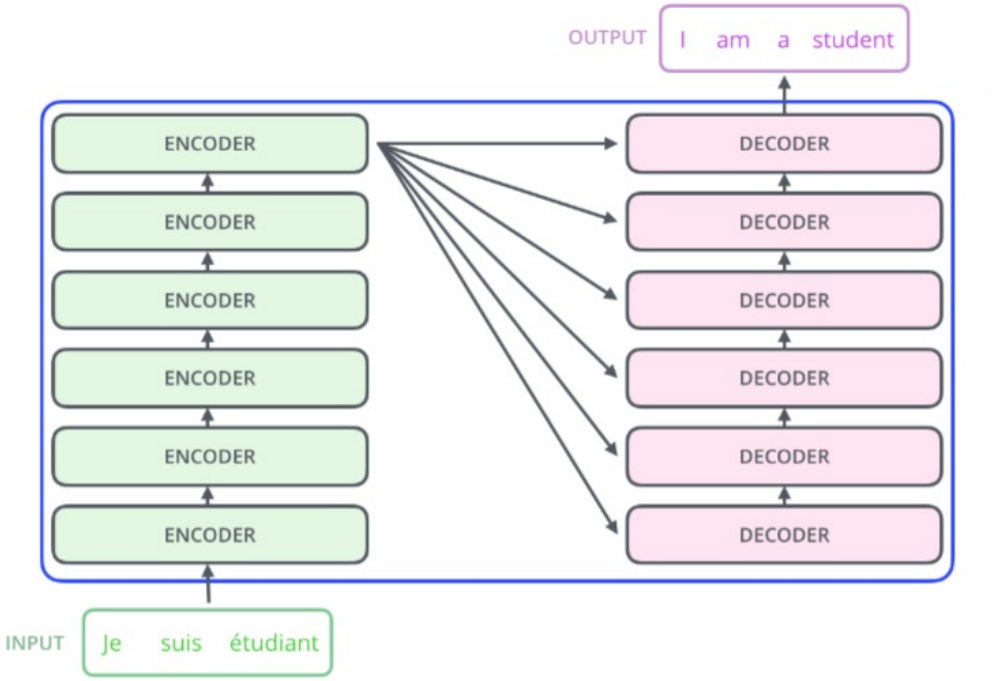


Рисунок 4.2 – Изображение наборов кодировщиков и декодировщиков [27]

На 4.2 изображен пример подсистем, состоящих из наборов 6 кодировщиков и 6 декодировщиков.

Каждый кодировщик, в свою очередь, состоит из двух уровней, называемых слоями [28]:

– слой, выявляющий логические взаимосвязи слов внутри одного предложения с помощью механизма, называемого внутренним вниманием, который выполняет линейные преобразования над входными векторами и передает их на следующий слой;

– слой, который позволяет устранять двусмысленности значений слов, используя контекст, добавленный в вектор в результате работы предыдущего слоя. Данный слой называют слоем нелинейных преобразований [29].

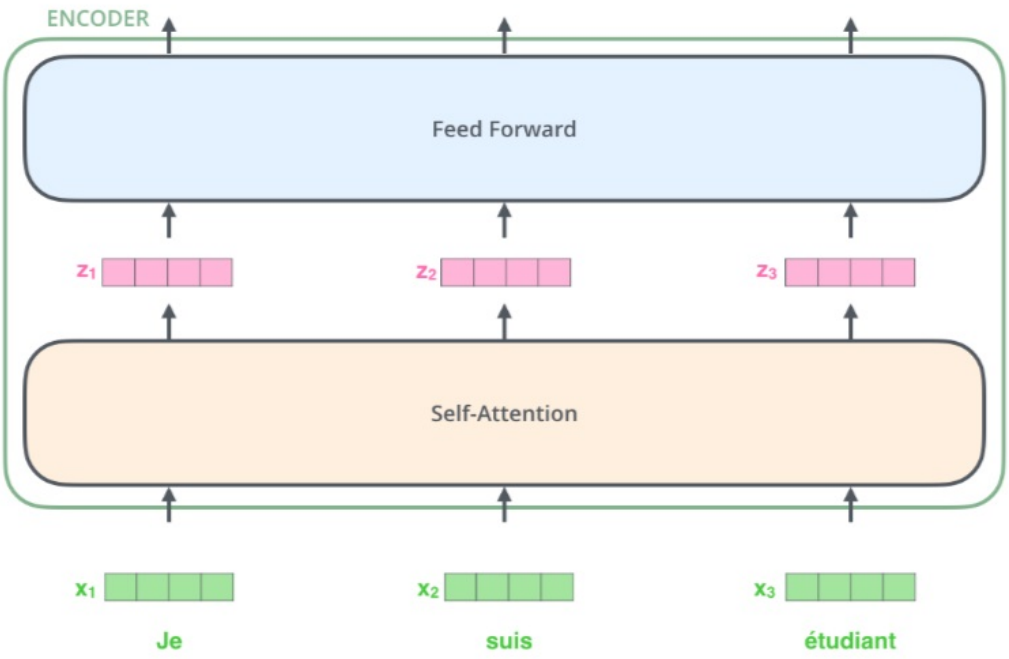


Рисунок 4.3 – Схема устройства кодировщика [27]

На рисунке 4.3 внизу кодировщика изображен слой с механизмом внутреннего внимания, вверху — слой нелинейных преобразований.

## 4.4 Механизм внутреннего внимания

Работа данного механизма описывается функцией, принимающей на вход один вектор, называемый текущим вектором, и набор всех векторов, включая текущий, полученных из исходного предложения. Данная функция добавляет к текущему вектору информацию о его связи с остальными векторами. Чтобы определить связь, необходима функция сравнения векторов, входными данными которой являются пара ключей, характеризующих сравниваемые векторы. Ключ, характеризующий текущий вектор, называют вектором запроса, а ключ, характеризующий вектор, с которым сравнивают текущий вектор — вектором ключа. Также необходимо уметь извлекать информацию об исходном токене вектора, называемую вектором значения, она будет использована для добавления информации о связи в текущий вектор. Получение отдельно вектора запроса может быть полезно, когда, например, есть необходимость исключения сильной связи между одинаковыми векторами [25].

Таким образом, для работы данного механизма необходимо уметь извлекать из вложения следующую информацию:

– вектор запроса;

– вектор ключа;

– вектор значения.

Одним из результатов обучения модели Трансформера являются матрицы $W^Q$, $W^K$, $W^V$, которые нужны для извлечения из вложения информации о запросе, ключе и значении соответственно.

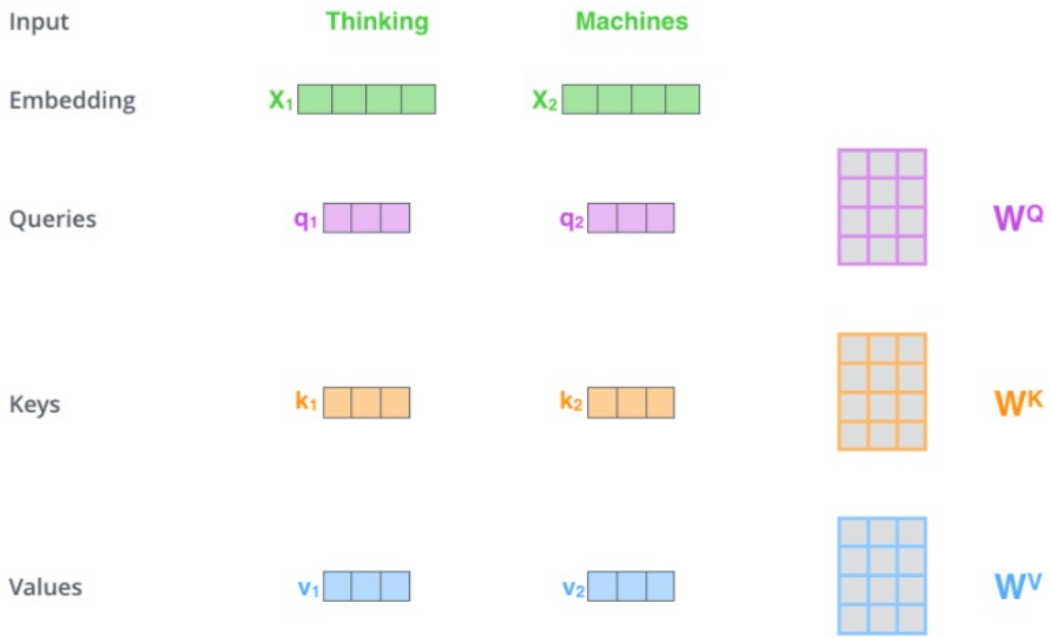


Рисунок 4.4 – Векторы запросов, ключей и значений [27]

На рисунке 4.4 изображены векторы запросов ($q_1$ и $q_2$), ключей ($k_1$ и $k_2$) и значений ($v_1$ и $v_2$), полученные в результате умножения векторов-строк ($X_1$ и $X_2$) на матрицы $W^Q$, $W^K$, $W^V$.

Механизм внутреннего внимания может быть описан функцией [30]:

$$attention(q, k, v) = \sum_{i=1}^{N} softmax[\frac{similarity(q, k_i)}{\sqrt{d_k}}] \cdot v_i, \text{где} \qquad (4.2)$$

*$q$ — вектор запроса, $k$ — массив векторов ключей всех вложений, $v$ — массив векторов значений всех вложений, $N$ — количество вложений в предложении, $similarity$ — функция, возвращающая значение, характеризующее схожесть векторов.*

На практике функция схожести $similarity$ обычно представлена векторным скалярным произведением [25].

$$softmax(s) = \frac{expexp(s_i)}{\sum_j s_j}. \tag{4.3}$$

Результат скалярного произведения может варьироваться в большом промежутке значений, поэтому для получения корректных результатов работы функции $softmax$ необходимо масштабировать результат скалярного произведения умножением на константу $\frac{1}{\sqrt{d}}$, где $d$ — это размерность вектора-ключа и вектора-запроса [27].

Таким образом, механизм внимания для каждого вектора из последовательности выполняет действия [27]:

1) формирование вектора запроса с помощью умножения матрицы $W^Q$ на данный (внешний) вектор;

2) получение в векторном виде информации о связи с каждым вектором из последовательности путем выполнения следующих шагов:

   2.1) создание с помощью умножения матриц $W^K$ и $W^V$ на текущий вектор двух новых векторов: ключа, для нахождения схожести с вектором запроса внешнего вектора, и значения, связанного непосредственно с вложением текущего вектора;

   2.2) вычисление скалярного произведения ключа каждого входного вектора и запроса текущего вектора, называемое коэффициентом схожести;

   2.3) масштабирование коэффициента схожести с помощью деления на $d_k$;

2.4) вычисление $softmax$ от масштабированного коэффициента, называемое долей вклада значения;

2.5) умножение значения на его долю вклада.

3) сложение всех полученных векторов.

Механизм внимания может обрабатывать векторы независимо, поэтому, для ускорения вычислений, из запросов, ключей и значений, полученных из вложений, формируют матрицы, давая тем самым возможность применять алгоритмы матричного умножения, что позволяет ускорить вычисления [25].

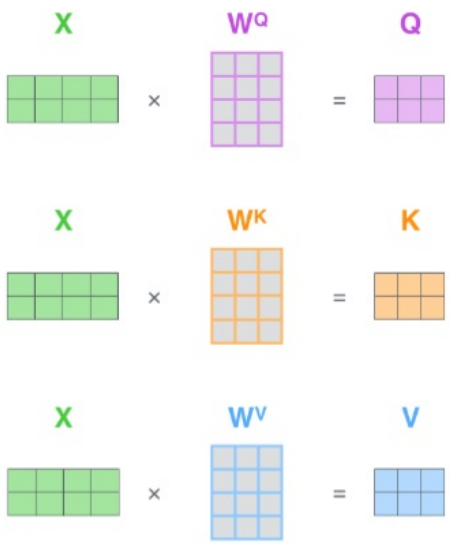


Рисунок 4.5 – Вычисление матриц запросов, ключей и значений [27]

На рисунке 4.5 изображена матрица $X$, сформированная из всех векторов входной последовательности $x$. Матрицы $Q$, $K$ и $V$, хранящие запросы, ключи и значения соответствующих векторов, получаются в результате умножения матрицы $X$ на матрицы $W^Q$, $W^K$ и $W^V$.

Таким образом, в матричной форме механизм внимания может быть представлен следующей формулой [30]:

$$attention(Q, K, V) = softmax(\frac{Q \cdot K^T}{\sqrt{d}}) \cdot V. \tag{4.4}$$

На практике используются несколько блоков внимания, называемых головами, а соответствующий механизм называют механизмом внутреннего

внимания со множеством голов. Каждая голова может быть использована для нахождения различных зависимостей между разными частями предложения [25].

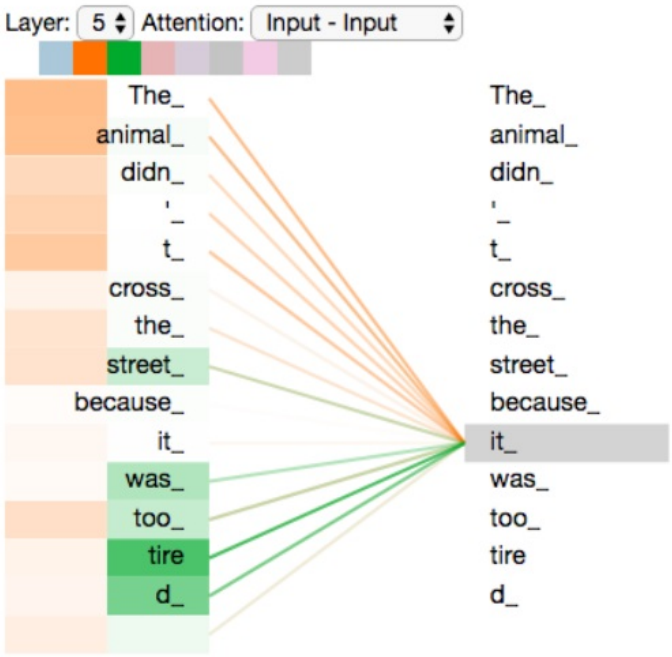


Рисунок 4.6 – Пример работы механизма внутреннего внимания со множеством голов [27]

На рисунке 4.6 отображена зависимость между токеном *it* и остальными токенами в предложении, причем более контрастному цвету соответствует более сильная связь. Одна из голов механизма внимания, соответствующая первому столбцу, определила, что местоимение *it* связано с артиклем *the* и существительным *animal*. Другая голова, соответствующая второму столбцу, определила связь того же местоимения со сказуемыми *was* и *tired*. Таким образом можно предположить, что одна из голов ищет связи токена с артиклями и существительными, а другая — связи со сказуемыми. Первый кодировщик из набора может определить зависимости только между отдельными токенами исходного предложения, второй — между парами векторов, заключающих в себе информацию о соответствующих токенах и их связи с остальными токенами предложения и так далее. Таким образом, наличие набора кодировщиков позволяет осуществить поиск сложных взаимосвязей между частями предложения [27].

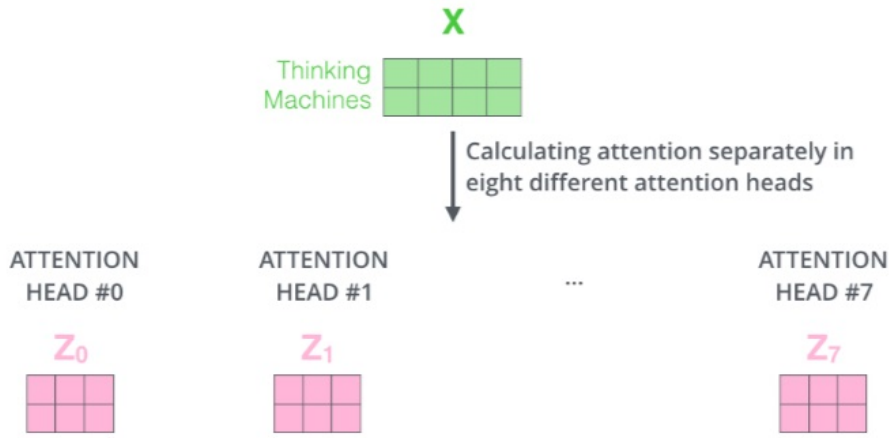


Рисунок 4.7 – Пример работы механизма внимания со множеством голов [27]

На рисунке 4.7 представлен процесс добавления в векторам информации о взаимосвязи с другими векторами.

Каждая голова может быть представлена как функция внимания [30]:

$$head_i(x) = attention(W_i^Q x, W_i^K x, W_i^V x), \text{ где} \tag{4.5}$$

$W_i^Q$ — *матрица i-й головы для получения вектора запроса,* $W_i^K$ — *матрица i-й головы для получения вектора ключа,* $W_i^V$ — *матрица i-й головы для получения вектора значения.*

Головы создают промежуточные матрицы, строками которых являются векторы, хранящие дополнительно информацию о связи текущего вектора с другими векторами.

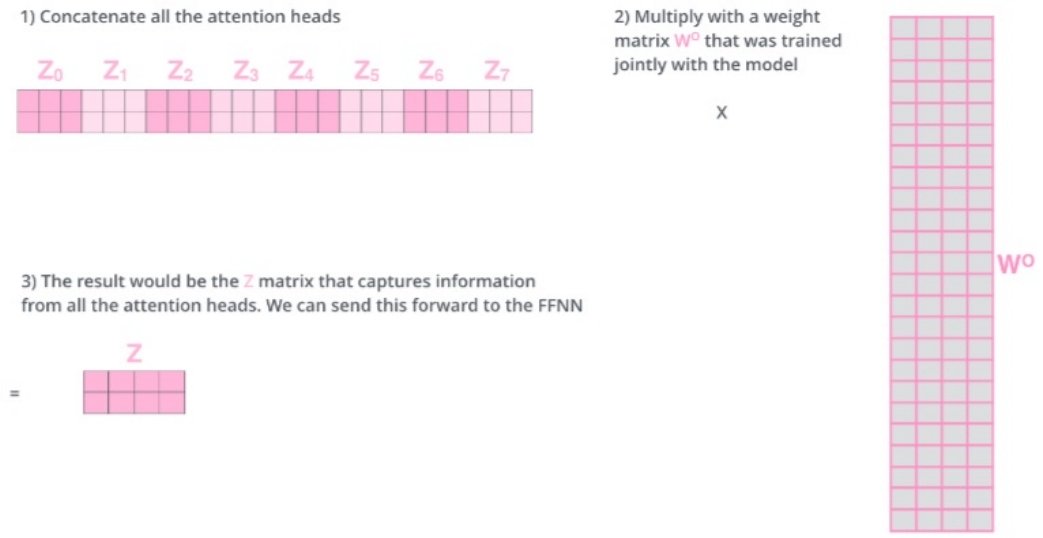


Рисунок 4.8 – Объединение значений промежуточных матриц [27]

Для объединения этих промежуточных матриц используется функция *concat*, которая объединяет матрицы вдоль левого края, получая тем самым матрицу размером $N \times (M \cdot d)$, где $N$ — количество вложений в предложении, $M$ — количество голов у механизма внимания. Эта объединенная матрица умножается на $W_0$ — матрицу, размерностью $(M \cdot d) \times d_v$, где $d_v$ – это размерность вектора-значения.

Механизм внутреннего внимания со множеством голов может быть представлен следующей формулой:

$$multihead(x) = W_0 \cdot concat[head_1(x), head_2(x), \ldots, head_n(x)]. \qquad (4.6)$$

## 4.5 Декодирующий компонент

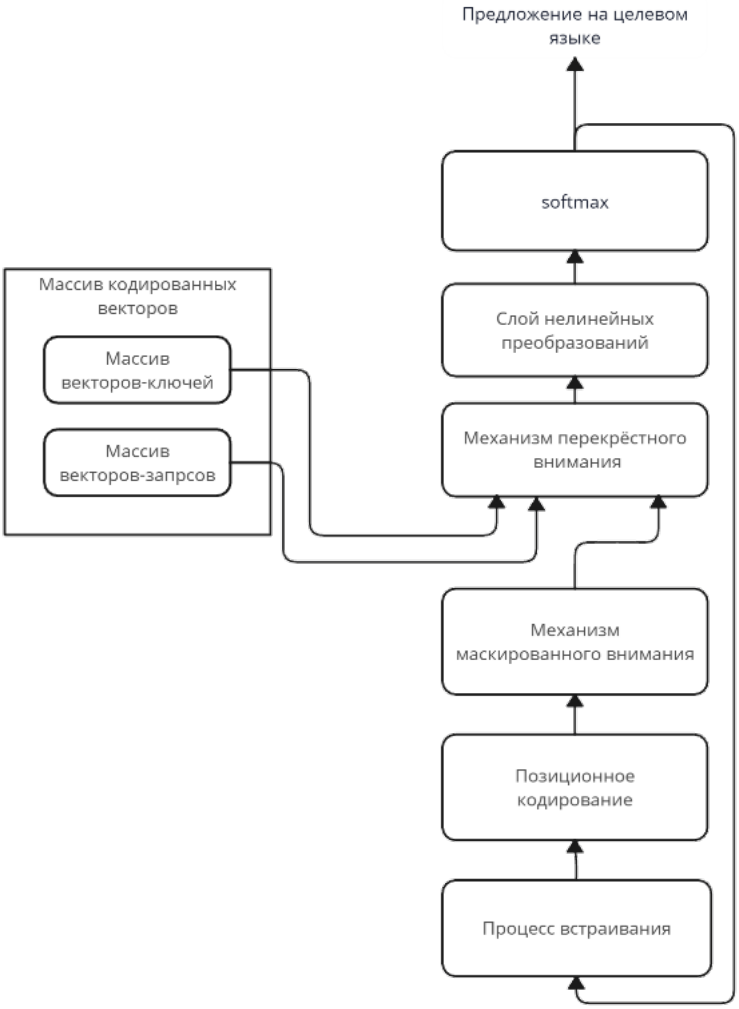


Рисунок 4.9 – Схема декодирующего компонента [27]

На рисунке 4.9 изображен декодирующий компонент, включающий в себя два механизма внимания [25]:

– необходимый для учета уже переведенной части предложения, называемый механизмом маскированного внимания;

– связывающий векторы ключей и значений выходной последовательности кодирующего компонента и векторов запроса, полученных из вектора, являющегося результатом работы механизма маскированного внимания. Данный механизм называется механизмом перекрестного внимания.

Механизм маскируемого внимания можно представить с помощью следующей формулы [30]:

$$maskedAttention(Q, K, V) = softmax(\frac{Q \cdot K^T + M}{\sqrt{d}}) \cdot V, \text{ где} \qquad (4.7)$$

*$M$ — матрица, называемая маскирующей, элементы которой заполняются значениями* 0 *и* $-\infty$ *таким образом, чтобы значение внутри функции* $softmax$ *оставалось неизменным в том случае, если позиция вставки не превышает позиции последнего слова в переведенном предложении и равно* 0 *в обратном случае.*

Необходимость маскирующей матрицы $M$ в формуле 4.7 следует из того, что количество строк в матрице, содержащей векторы-ключи и векторы-значения, соответствующие исходному предложению, должны совпадать с количеством строк матрицы, содержащей векторы-запросы в матрице, соответствующей векторам целевого предложения, однако в процессе перевода некоторое количество строк из матрицы векторов-запросов будет содержать мусорные значения, так как соответствующие слова еще не были переведены, поэтому необходимо обеспечить отсутствие влияния данных векторов на конечный результат [30].

Механизм перекрестного внимания схож с обычным механизмом внимания, за исключением того, что анализируются связи между двумя предложениями.

Результатом работы декодирующего компонента является вектор рациональных чисел. Слой декодирующего компонента, входом которого является выход последнего декодера, называется линейным слоем. Этот слой переводит исходный вектор размерностью $d_{model}$ в вектор с размерностью, равной количеству элементов в словаре. В каждой ячейке полученного вектора будет находиться оценка, пропорциональная вероятности генерации следующего слова, находящегося на позиции в словаре, соответствующей позиции ячейки вектора.

Последний слой, представляющий из себя функцию $softmax$, переводит оценки, полученные в результате работы предыдущего слоя, в вероятности, то есть сумма элементов результирующего вектора равна единице.

Результатом работы декодирующего компонента на последнем шаге

будет элемент словаря, соответствующий позиции элемента вектора последнего слоя со значением, наиболее близким к единице [27].

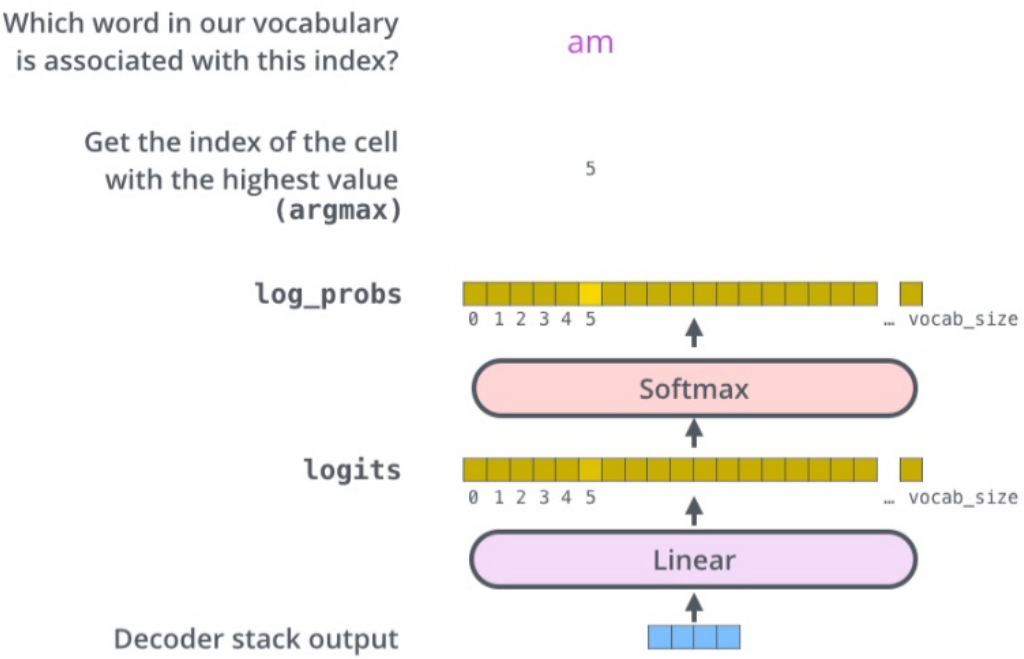


Рисунок 4.10 – Пример результата работы декодирующего компонента [27]

На рисунке 4.10 изображен вектор $logits$, являющийся результатом работы последнего декодера. Этот вектор поступает на вход линейному слою, генерирующему вектор размерностью, равной количеству слов в словаре. Далее, этот вектор проходит через функцию $softmax$, преобразуя его в вектор $log_probs$. Наибольшее значение в векторе $log_probs$ имеет элемент с индексом 5, соответственно, из словаря целевого языка модели берется элемент с таким же индексом, то есть, слово $am$. Данное слово является результатом работы декодирующего компонента на текущем шаге. Этот процесс будет происходить, пока декодирующим компонентом не будет встречен вектор, связанный с токеном, обозначающим конец предложения, что будет означать окончание процесса перевода.

# 5 Сравнение методов машинного перевода

Критерии:

– Качество перевода;

– Временная сложность.

## 5.1 Оценка качества машинного перевода

Одной из самых популярных и доступных на данный момент метрик оценки качества перевода является BLEU (Bilingual Evaluation Understudy). В ходе оценки качества измеряется степень близости МП к одному или нескольким переводам, выполненным человеком, при помощи числовой метрики.

В BLEU не учитывается синтаксис и порядок слов (но определяются более длинные совпадающие n-граммы) [31]. Метрику BLEU можно представить в виде функции:

$$BLEU(Candidates) = BP * \exp(\sum_{n=1}^{N} \omega_n \ln p_n(Candidates)), \text{ где} \tag{5.1}$$

*Candidates — текст, полученный в результате МП;* $\omega_n$ *— веса, сумма которых равна единице; BP (brevity penalty) — штраф за длину перевода:*

$$BP(n) = \begin{cases} 1 & \text{, если } c > r \\ exp(1 - \frac{r}{c}) & \text{, если } c \leq r \end{cases}, \text{ где} \tag{5.2}$$

*c — количество слов в машинном переводе; r — количество слов в эатлонном переводе;* $p_n$ *— оценка вероятности для всего текста, полученного в результате МП:*

$$p_n(Candidates) = \frac{\sum\limits_{C \in \{Candidates\}} \sum\limits_{\text{n-gram} \in C} Count_{clip}(\text{n-gram})}{\sum\limits_{C' \in \{Candidates\}} \sum\limits_{\text{n-gram}' \in C'} Count(\text{n-gram}')}, \text{ где} \tag{5.3}$$

$Count_{clip}(n\text{-}gram)$ *— функция для поиска в тексте, полученном в результате МП, количества n-грамм, не превышающего количества этих*

*n-грамм в эталонных текстах:*

$$Count_{clip}(\text{n-gram}) = min(Count(\text{n-gram}), Max\_Ref\_Count), \text{ где} \quad (5.4)$$

*$Count(n\text{-}gram)$ — функция для поиска в тексте, полученном в результате МП, количества n-грамм, $Max\_Ref\_Count$ — количество n-грамм n-gram в эталонном переводе.*

Обычно, в BLUE используются n-граммы длины до 4 включительно ($N = 4$) и $\omega_n = \frac{1}{N}$ [32].

$$\begin{aligned} exp(\sum_{n=1}^{N} \omega_n \log p_n) &= \Pi_{n=1}^{N} exp(\omega_n \log p_n) = \\ &= \Pi_{n=1}^{N} [exp(\log p_n)]^{\omega_n} = \\ &= \Pi_{n=1}^{N} p_n^{\omega_n} \\ &\in [0, 1]. \end{aligned} \quad (5.5)$$

Согласно формулам 5.4 и 5.5, левый и правый множители в функции $BLEU$ имеют значения в пределах от 0 до 1 включительно, поэтому значения функции $BLEU$, равной произведению этих множителей, также варьируются в пределах от 0 до 1, что также можно представить в виде процентного соотношения.

Таблица 5.1 – Оценка качества машинного перевода по оценке BLEU для языковой пары English-French [33]

| Метод | Оценка |
|---|---|
| СМП | 27.3% |
| RNN | 34.54% |
| LSTM | 34.8% |
| Transformer | 41.5% |

## 5.2 Оценка временной сложности

Таблица 5.2 – Оценка временной сложности. $n$ — количество слов в переводимом предложении, $d$ — количество слоев кодировщиков модели [25; 34].

| Метод | Временная сложность |
|---|---|
| СМП | $O(n \cdot d)$ |
| RNN | $O(n \cdot d^2)$ |
| LSTM | $O(n \cdot d^2)$ |
| Transformer | $O(n^2 \cdot d)$ |

## 5.3 Вывод

Согласно выбранным критериям оценивания качества перевода и приведенным измерениям, более предпочтительной оказалась модель Трансформер.

В соответствии с выбранными критериями оценки временной сложности, можно сказать, что в случае, когда среднее количество слов в предложении превышает количество слоев кодировщиков, то Трансформер является менее эффективным по времени, чем RNN и LSTM. В противном случае, если количество кодировщиков превышает среднее количество слов в предложении, то Трансформер является более эффективным по времени, чем RNN и LSTM. Статистический машинный перевод имеет временную сложность меньшую, чем у остальных перечисленных системы.

# ЗАКЛЮЧЕНИЕ

В ходе данной работы были решены следующие задачи:

– проведен анализ предметной области машинного перевода европейских языков;

– описаны основные подходы решения задачи машинного перевода европейских языков;

– сформулированы критерии сравнения применяемых методов и выполнена их классификация.

Цель работы, а именно проведение анализа существующих методов машинного перевода европейских языков, также была достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Кибрик А. Е. Язык // Языкознание. Большой энциклопедический словарь. — М.: Большая Российская энциклопедия, 1998. — С. 605.
2. Европейские языки [Электронный ресурс]. — Режим доступа: https://budushiypoliglot.blogspot.com/2020/07/yevropeyskiye-yazyki.html (дата обращения: 10.10.2023).
3. Статистическая система машинного перевода [Электронный ресурс]. — Режим доступа: https://elib.bsu.by/bitstream/123456789/92831/1/58.pdf (дата обращения: 08.11.2023).
4. Машинный перевод [Электронный ресурс]. — Режим доступа: https://fsc.bsu.by/wp-content/uploads/2015/12/Mashinny-j-perevod-konspekt-lektsij.pdf (дата обращения: 10.10.2023).
5. Shiwen Y., Xiaojing B. Rule-based machine translation //Routledge encyclopedia of translation technology. — Routledge, 2014. — С. 186-200.
6. Головко Д. Р. Особенности и виды машинного перевода //Вестник Московского информационно-технологического университета–Московского архитектурно-строительного института. — 2020. — №. 4. — С. 26-27.
7. Chéragui M. A. Theoretical Overview of Machine translation //ICWIT. — 2012. — С. 160-165.
8. Добров А. В. Глава 2. Компьютерный синтаксис // Прикладная и компьютерная лингвистика. — 2017. — С. 35-58.
9. Poibeau T. Machine translation. — MIT Press, 2017. — С. 25-32.
10. Richens R. H. Interlingual machine translation //The Computer Journal. — 1958. — Т. 1. — №. 3. — С. 144-147.
11. *Koehn P.* Statistical Machine Translation. — The Edinburgh Building, Cambridge CB2 8RU, UK : Cambridge University press, 2010.
12. Рекуррентная нейронная сеть (RNN): виды, обучение, примеры [Электронный ресурс]. — Режим доступа: https://neurohive.io/ru/osnovy-data-science/rekurrentnye-nejronnye-seti/ (дата обращения: 10.10.2023).

13. Что такое слои встаивания в нейронной сети? [Электронный ресурс]. — Режим доступа: https://www.baeldung.com/cs/neural-nets-embedding-layers (дата обращения: 10.10.2023).

14. Нейронные сети, или Как обучить искусственный интеллект [Электронный ресурс]. — Режим доступа: https://ii.org.ru/neyronnye-seti-ili-kak-obuchit-iskuss/ (дата обращения: 10.10.2023).

15. Скрытые слои нейронной сети [Электронный ресурс]. — Режим доступа: https://spravochnick.ru/informatika/skrytye_sloi_neyronnoy_seti/ (дата обращения: 10.10.2023).

16. Понимание сетей LSTM [Электронный ресурс]. — Режим доступа: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (дата обращения: 10.10.2023).

17. LSTM - долговременная кратковременная память [Электронный ресурс]. — Режим доступа: https://questu.ru/articles/281163/ (дата обращения: 10.10.2023).

18. Функции активации [Электронный ресурс]. — Режим доступа: https://robotdreams.cc/blog/327-funkciji-aktivaciji-stupinchasta-liniyna-sigmojida-relu-ta-tanh#item1 (дата обращения: 10.10.2023).

19. LSTM — долгая краткосрочная память [Электронный ресурс]. — Режим доступа: https://proproprogs.ru/neural_network/lstm-dolgaya-kratkosrochnaya-pamyat (дата обращения: 10.10.2023).

20. Sameen M. A Survey on Document-level Neural Machine Translation: Methods and Evaluation, 2021.

21. Felix Stahlberg. Neural Machine Translation: A Review, 2020.

22. Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate, 2015.

23. RAJ DABRE. A Survey of Multilingual Neural Machine Translation, 2020.

24. Макарова Д. А., Шибанова А. Д. Искусственные нейронные сети, 2019.

25. Attention Is All You Need, Ashish Vaswani, Noam Shazeer, Niki Parmar, 2017.

26. Thierry P. Machine Translation, 2017.

27. Jay Alammar. The Illustrated Transformer, 2020.

28. Anni Burchfiel. What is NLP (Natural Language Processing) Tokenization, 2020.

29. Damien S. Understanding BERT Transformer: Attention isn't all you need, 2019.

30. Pascal Poupart, конспект лекций CS480/680 — Introduction to Machine Learning, 2019.

31. Автоматическая оценка качества машинного перевода научного текста [Электронный ресурс]. — Режим доступа: https://vestnik-mgou.ru/Articles/Doc/15049 (дата обращения: 10.10.2023).

32. BLEU: a Method for Automatic Evaluation of Machine Translation [Электронный ресурс]. — Режим доступа: https://aclanthology.org/P02-1040.pdf (дата обращения: 10.10.2023).

33. Machine Translation on WMT2014 English-French [Электронный ресурс]. — Режим доступа: https://paperswithcode.com/sota/machine-translation-on-wmt2014-english-french (дата обращения: 10.10.2023).

34. Franz J. Statistical Machine Translation: From Single-Word Models to Alignment Templates, 2003.