МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)
01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки, специальности)
Системное программирование и компьютерные технологии
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему **Исследование алгоритмов обработки естественного языка с применением машинного обучения**

Студент	Д.С. Саксонов	
	(И.О. Фамилия)	(личная подпись)
Руководитель	С.В. Мкртычев	
	(И.О. Фамилия)	(личная подпись)
Консультанты	Н.В. Андрюхина	
	(И.О. Фамилия)	(личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)
«_____»_____20_____г.

Тольятти 2019

АННОТАЦИЯ

Объектом исследования бакалаврской работы является процесс получения представлений признаков при обработке естественного языка.

Целью данной работы является выявление факторов, влияющих на производительность моделей обработки естественного языка.

В исследовании выдвигается гипотеза о том, что производительность моделей обработки естественного языка зависит от реализации двунаправленности языковой модели.

В результате проведения экспериментов, были получены результаты, подтверждающие выдвигаемую гипотезу. Так, было выявлено, что представления слов, моделированные двунаправленной рекуррентной нейронной сетью, уступают в эффективности представлениям, полученным с применением двунаправленной модели кодировщика-трансформера.

Также были получены новые данные о влиянии использования предварительно обученных представлений на регуляризацию моделей.

В первом разделе определяются современные методы обработки естественного языка на основе машинного обучения. В нем также формулируется задача исследования, основанная на выводах о современном состоянии предметной области.

Второй раздел посвящен математическому моделированию базовой модели исследования. В нем представлены результаты моделирования и сравнительный анализ производительности разработанной модели с существующими решениями.

В третьем разделе описаны экспериментальные модели и приведены результаты экспериментов.

Работа состоит из введения, трех разделов и заключения. Количество рисунков в работе – 6; количество таблиц – 5. Объем работы составляет 48 страниц. Список литературы содержит 64 наименования.

ABSTRACT

The topic of the given graduation work is “Research on machine learning algorithms for natural language processing”.

The research aims to reveal factors impacting performance of a natural language processing (NLP) models.

The object of the research is a process of obtaining feature representations for usage in such models. The subject is a definition of a context in feature (namely word) representation models.

Main part of the graduation work is dedicated to the issues of representing words numerically (using word vectors) while maintaining its meaning relative to the whole vocabulary of a model (i.e. embedding it in a vector space).

Distributed word representations are able to capture different features of words (semantic, syntactic or even sentiment) out of raw text corpus without any comprehensive pre-modelling. Most of the time those features are task independent which makes them ideal for language modelling. This makes the task of obtaining such representations one of the most important steps in building a high quality NLP model.

However, there are many approaches being researched today, each showing state-of-the-art results for many NLP tasks. The scope of this research is on two different approaches for modelling bidirectionality in word representations.

The results of the study showed that the recurrent approach to modelling bidirectionality is inferior at capturing semantic meaning of words to the transformer encoding based approach. The dependency of a representation model size to its performance on a certain task's capacity was also revealed as a result of experiments.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. За последние несколько лет нейронные сети приобрели статус крайне эффективных моделей машинного обучения, показывающих передовые результаты в таких областях, как распознавание изображений и обработка речи. В последнее время модели на основе нейронных сетей стали применяться к текстовым данным естественного языка, так же показывая многообещающие результаты.

До недавнего времени многие системы и методы обработки естественного языка рассматривали слова как атомарные единицы – понятие об отношениях между словами отсутствовало, поскольку они представлялись в виде индексов в словарях. Выбор в пользу данного подхода имеет несколько веских причин – простота, надежность и наблюдение того факта, что простые модели, обученные на больших объемах данных, превосходят сложные системы, обученные на меньшем количестве данных. Примером является популярная модель N-грамм, используемая для статистического языкового моделирования – сегодня можно обучить N-граммы практически для всех доступных данных (триллионы слов [29]).

С развитием технологий машинного обучения в последние годы стало возможным обучать более сложные модели на гораздо больших наборах данных. Вероятно, самым концептуально большим шагом в области обработки естественного языка стал переход к использованию распределенных представлений слов [43].

Предварительно обученные (векторные) представления слов [13] – ключевой компонент многих современных моделей обработки естественного языка. Однако, получение высококачественных представлений является сложной задачей. Они должны моделировать как сложные характеристики применения слов (например, синтаксис и семантика), так и то, как эти применения варьируются в зависимости от лингвистического контекста (например, для моделирования многозначности).

На сегодняшний день также существуют различные подходы к получению представлений, которые позволяют моделировать не только семантическую составляющую слова, но и контекст, в котором слово употребляется, позицию слова в предложении и некоторые другие лингвистические характеристики.

Обработка естественного языка – сфера, которая сегодня продолжает активно развиваться, принося новые архитектуры моделей [7] и подходы к сбору и обработке данных для всех областей машинного обучения.

Все вышеперечисленные аргументы определяют актуальность темы исследования.

Проблема исследования: эффективность подходов к созданию представлений признаков при обработке естественного языка.

Объект исследования: процесс получения представлений признаков при обработке естественного языка.

Предмет исследования: определение контекста в моделях представлений слов для обработки естественного языка.

Цель исследования: выявление факторов, влияющих на производительность моделей обработки естественного языка посредством проведения экспериментов с различными методами представления признаков.

Задачи исследования:

1. Раскрыть основные задачи обработки естественного языка.
2. Определить современные методы получения представлений признаков при обработке естественного языка.
3. Провести серию экспериментов с выбранными методами на задаче семантического разбора.
4. Провести анализ полученных результатов и выявить факторы, влияющие на производительность моделей обработки естественного языка.

В первом разделе определяются современные методы обработки естественного языка на основе машинного обучения. В нем также

формулируется задача исследования, основанная на выводах о современном состоянии предметной области.

Второй раздел посвящен математическому моделированию базовой модели исследования. В нем представлены результаты моделирования и сравнительный анализ производительности разработанной модели с существующими решениями.

В третьем разделе описаны экспериментальные модели и приведены результаты экспериментов.

В заключении приведены основные результаты и выводы проведенного исследования.

1 ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА

1.1 Архитектуры нейронных сетей

Нейронные сети являются крайне эффективными моделями машинного обучения. Архитектуры, применяемые для задач обработки естественного языка, можно разделить на две группы: сети прямого распространения и рекуррентные/рекурсивные сети. Сети прямого распространения включают в себя полносвязные сети, такие как многослойный перцептрон, а также сети со сверточными и агрегирующими (англ. pooling) слоями.

Полносвязные сети прямого распространения являются нелинейными обучаемыми моделями, которые, в большинстве случаев, могут использоваться во всех задачах, где возможно применение линейных обучаемых моделей. Эти задачи включают в себя задачи бинарной и многоклассовой классификации, а также более сложные задачи структурированного предсказания. Нелинейность модели, а также возможность использования предварительно обученных представлений слов (признаков), приводит к крайне высокой точности классификации. В ряде работ [14, 45] удалось получить лучшие результаты синтаксического анализа, просто заменив линейную модель синтаксического анализатора на полносвязную нейронную сеть прямого распространения. Применение сетей прямого распространения в качестве классификатора (обычно с использованием предварительно обученных представлений слов) также дает прирост в производительности для таких задач, как отслеживание состояния диалога [21], предварительное упорядочивание для статистического машинного перевода [17] и языковое моделирование [11].

Сети со сверточными и агрегирующими слоями эффективны в задачах классификации, когда ожидается наличие локальных признаков, указывающих на принадлежность классу, но эти признаки могут проявляться в различных участках структуры входных данных. Например, в задаче классификации документов отдельная ключевая фраза (или n-грамма) может помочь в определении тематики документа [26]. Такие модели эффективны, когда верно

предположение о том, что определенные последовательности слов являются хорошими индикаторами темы документа и возможно пренебречь их положением в этом документе. Сверточные и агрегирующие архитектуры показывают многообещающие результаты для многих задач, включая классификацию документов, обнаружение событий [36], генерирование ответов на вопросы [39] и другие задачи, в которых поиск локальных признаков является эффективным подходом [9, 49].

В обработке естественного языка зачастую приходится работать со структурированными данными произвольных размеров, такими как последовательности и деревья. Для решения большинства задач необходимо иметь возможность фиксировать закономерности в таких структурах или моделировать сходства между ними. В большинстве случаев это означает, что необходимо закодировать такую структуру в вектор с фиксированной шириной, который затем можно будет передать другой статистической модели для дальнейшей обработки. В то время как сверточные и агрегирующие архитектуры позволяют кодировать произвольные крупные элементы как векторы фиксированного размера, фиксируя их наиболее характерные особенности, они делают это, жертвуя большей частью информации о структуре. Рекуррентные и рекурсивные архитектуры, с другой стороны, позволяют вести работу с последовательностями и деревьями, сохраняя большую часть информации об их структуре. Рекуррентные сети [56] предназначены для моделирования последовательностей, в то время как рекурсивные сети [58] являются обобщением рекуррентных сетей, которые могут обрабатывать деревья.

Рекуррентные модели показывают высокие результаты для задач языкового моделирования [40], разметки частей речи [24], машинного перевода [48], грамматики зависимостей [47], сентиментального анализа, нормализации шумного текста [10], отслеживания состояния диалога [33] и генерации ответов [5].

Рекурсивные модели показывают передовые или около-передовые результаты для синтаксического анализа на основе грамматики составляющих [37], дискурсивного анализа [30], классификации семантических отношений [3], сентиментального анализа [22] и генерирования ответов на вопросы [6].

Исходя из вышесказанного можно сделать вывод о том, что для решения задач обработки естественного языка применимы общепринятые архитектуры моделей машинного обучения. Так, сверточные и агрегирующие сети широко распространены и являются основой для задач обработки изображений, а рекуррентные сети применимы ко всем задачам, связанным с обработкой последовательностей, в случае, когда необходимо сохранить информацию о структуре последовательности.

Ряд исследований [15, 38] показал, что на производительность моделей обработки естественного языка главным образом влияет подход к представлению признаков, которые поступают на вход в модели для конкретных задач. Этот факт позволил научному сообществу сосредоточиться на разработке оптимальных методов получения представлений признаков.

1.2 Задача получения представлений признаков

1.2.1 Плотные векторы представлений

Прежде чем обсуждать архитектуры сетей более подробно, важно обратить внимание на подходы к представлению признаков. На данный момент можно представить нейронную сеть прямого распространения как функцию $NN(x)$, которая принимает на вход d_{in}-мерный вектор x и возвращает d_{out}-мерный вектор. Функция зачастую используется в качестве классификатора, присваивая вводу x оценку принадлежности к одному или нескольким классам в d_{out}. Функция может быть сложной и практически всегда является нелинейной. В задачах обработки естественного языка ввод x кодирует такие признаки, как слова, метки частей речи или другую лингвистическую информацию. Самым концептуально большим шагом при переходе от

линейных моделей с разреженным вводом к моделям на основе нейронных сетей – это прекращение представления каждого признака в виде уникального измерения (метод one-hot encoding) и вместо этого представление признаков в виде плотных векторов. Таким образом, каждый ключевой признак помещается в d-мерное пространство и представляется в виде вектора в этом пространстве. Такие представления (векторные представления каждого ключевого признака) могут быть обучены как любые другие параметры функции NN. На рисунке 1.1 показаны оба подхода к представлению признаков.

Значения элементов вектора ввода для каждого признака являются параметрами модели, которые должны быть обучены вместе с остальными компонентами сети.

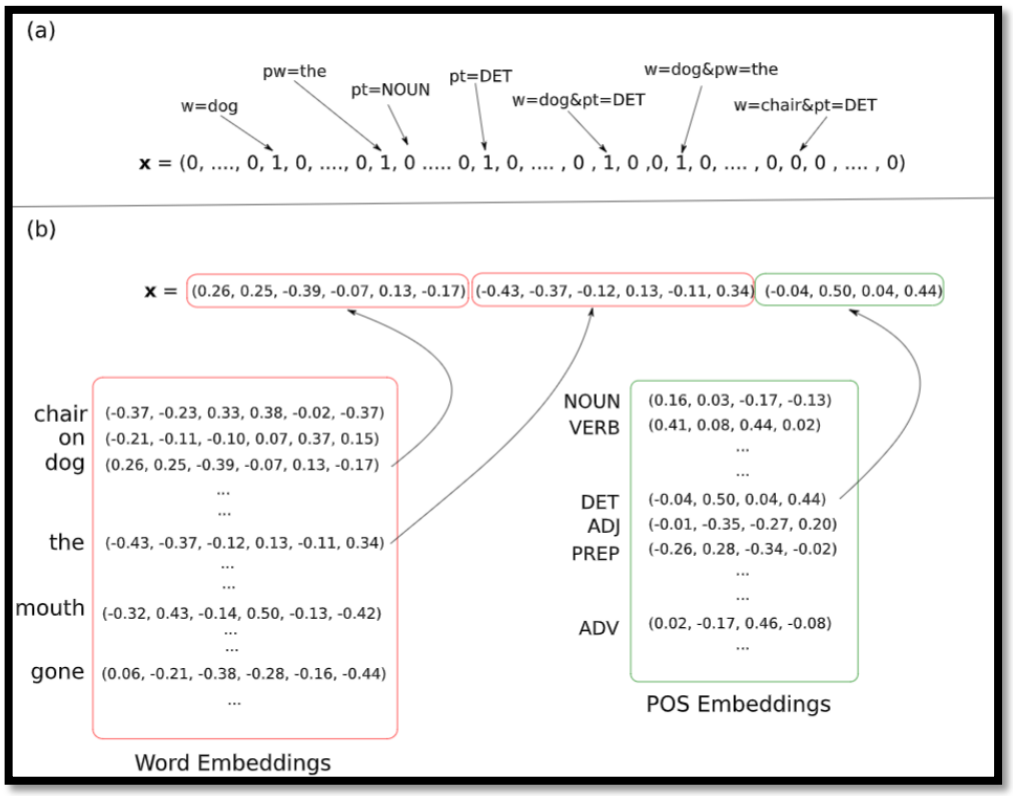

Рисунок 1.1 – Разреженные и плотные представления признаков. Два способа кодировки информации: *текущее слово – “dog”; предыдущее слово – “the”; предыдущий pos-tag (метка части речи) – “DET”*.

На рисунке 1.1 a - разреженный вектор признаков. Каждое измерение представляет признак. Комбинации признаков имеют собственные измерения. Значения признаков – бинарные. Размерность очень высокая. b – плотный вектор признаков. Каждый ключевой признак представлен в виде вектора. Каждый признак соответствует нескольким векторам ввода. Явная кодировка комбинаций признаков не нужна. Размерность низкая. Отношения признак-вектор получаются из таблицы представлений.

Общая структура для системы-классификатора при обработке естественного языка, основанная на нейронной сети прямого распространения выглядит следующим образом:

1. Выделяется множество ключевых лингвистических характеристик $f_1, \dots, f_k$, релевантных для предсказания принадлежности к классу.

2. Для каждого признака f_i получается соответствующий вектор $v(f_i)$.

3. Векторы комбинируются (при помощи конкатенации, суммы или комбинации обеих операций) в вектор ввода x.

4. На вход нелинейному классификатору поступает x.

Главным изменением в вводе, таким образом, является переход от разреженных представлений, в которых каждый признак представлен собственным измерением, к плотным представлениям, в которых каждый признак сопоставлен с определенным вектором. Другое отличие заключается в возможности выделять только ключевые признаки, не моделируя комбинации признаков заранее.

1.2.2 Векторные представления слов

Главным компонентом подхода к обработке естественного языка с применением нейронных сетей является использование векторных представлений (англ. embeddings) – представление каждого признака в виде

вектора в линейном векторном пространстве. Далее приводится описание общего подхода к получению векторных представлений слов.

При наличии достаточного количества данных для обучения с учителем, возможно обрабатывать представления признаков так же, как и другие параметры модели: инициализировать векторы представлений случайными значениями и позволить процессу обучения сети настроить их так, как это необходимо для минимизации функции потерь. Метод, использованный в имплементации word2vec [15, 13] состоит в инициализации векторных представлений слов равномерно выбранными случайными числами из диапазона $-\frac{1}{2d}, \frac{1}{2d}$ где d – количество измерений. Другой способ – использование инициализации Ксавьера [18] и инициализировать векторы равномерно выбранными случайными числами из диапазона $-\frac{\overline{6}}{\overline{d}}, \frac{\overline{6}}{\overline{d}}$.

По способу получения представлений, а также в зависимости от задачи, можно разделить процесс обучения на два типа: предварительное обучение с учителем для конкретной задачи и предварительное обучение без учителя.

Обучение с учителем эффективно в случаях, когда доступно достаточное количество маркированных данных. Так, например, если необходимо обучить модель для решения задачи А, для которой имеется ограниченное количество маркированных данных (например, задача синтаксического разбора), но есть вспомогательная задача Б (например, разметка частей речи), для которой имеется гораздо большее количество качественных маркированных данных, возможно провести предварительное обучение модели для задачи Б, а затем использовать полученные векторы для обучения новой модели задаче А. В процессе обучения векторы представлений можно рассматривать как фиксированные или оптимизировать их вместе с другими параметрами модели.

Однако, в большинстве случаев подходящая вспомогательная задача с большим количеством маркированных данных отсутствует. В таких случаях, возможно обучение векторных представлений слов путем «обучения без учителя». Техника обучения в таком случае остается неизменной, такой же, как

если бы применялось обучение с учителем, за тем лишь исключением, что вместо использования заранее подготовленных маркированных данных, задача для обучения создается из необработанного текста и в реальном времени.

Ключевой смысл подходов к обучению без учителя заключается в предположении о том, что одинаковые по смыслу слова должны иметь одинаковые векторы представлений. Хотя «схожесть» слов трудно определить, и способ ее измерения зачастую зависит от решаемой задачи, современные подходы базируются на дистрибутивной гипотезе, которая говорит, что слова одинаковы (по смыслу), если они встречаются в одинаковых контекстах. Смысл всех подходов к обучению без учителя в обработке естественного языка состоит в создании задач, в которых целью является предсказание слова из имеющегося контекста или контекста из имеющегося слова.

1.3 Анализ существующих решений и задача исследования

Для заданных слова w и его контекста c, различные алгоритмы формулируют разные вспомогательные задачи. Во всех случаях, каждое слово представлено как d-мерный вектор, который инициализируется случайными значениями. Обучение модели для решения вспомогательных задач приводит к получению представлений, связывающих слова с контекстами, что в результате приводит к схожим векторам представлений для схожих слов.

1.3.1 Алгоритм Word2Vec

Подходы, основанные на языковом моделировании, такие как те, что были выбраны при создании алгоритмов Word2Vec и GloVe [38], используют вспомогательные задачи, заключающиеся в предсказании слова для заданного контекста. Решение таких задач состоит в моделировании условной вероятности $P(w|c)$.

Некоторые другие подходы сводят проблему к бинарной классификации. В дополнение множеству D пар слово-контекст добавляется множество D,

созданное из случайных слов и контекстов. Задача бинарной классификации, в таком случае, следующая: является ли данная пара (w, c) членом D или нет? Такие подходы, в свою очередь, различаются в том, как составляется множество D, какова структура классификатора и какова целевая функция, которая оптимизируется. Колберт и Уэстон [35] применяют метод бинарного ранжирования на основе маржи, обучая нейронную сеть прямого распространения для выделения верных пар (w, c) среди ошибочных. Напротив, в алгоритме Word2Vec реализована вероятностная версия этой задачи. Суть алгоритма заключается в обучении логбилинейной модели для предсказывания вероятности $P(w, c \in D|w, c)$ того, что пары относятся к корпусу, а не к случайной выборке.

Процесс же обучения модели Word2Vec выглядит следующим образом:

1. Для модели определяются следующие гиперпараметры: размерность представлений, количество итераций обучения, шаг обучения и, наиболее характеризующий данную модель параметр – «размер окна». Окно здесь – это слова, находящиеся по обе стороны от исходного слова и определяющие его контекст. Принцип работы окна показан на рисунке 1.2.

2. Веса модели инициализируются случайным образом.

3. На вход модели подается вектор, представляющий целевое слово (для которого должно быть получено векторное представление), закодированное методом унитарной кодировки.

4. Производится операция произведения вектора ввода и весов первого слоя. Первый слой в модели – матрица, в которой хранятся векторные представления (случайно инициализированные) для каждого слова.

5. Вторая операция – произведение полученного векторного представления на веса слоя вывода. Второй слой является логистическим классификатором.

6. Полученный вектор проходит через сигмоидальную софтмакс функцию, вычисляя таким образом вероятность принадлежности целевого слова к контексту (к каждому слову из контекста).

7. Последний шаг обучения – вычисление ошибки и применение метода обратного распространения ошибки. Так, веса первого слоя обновляются, приводя к минимизации целевой функции (1).

$$
\begin{aligned}
E &= -\log p(w_{O,1}, w_{O,2}, \dots, w_{O,c} | w_I) \\
&= -\log \prod_{c=1}^{C} \frac{\exp\ u_{c,j_c^*}}{\sum_{j'=1}^{V} \exp(u_{j'})} \\
&= -\sum_{c=1}^{C} u_{j_c^*} + C \cdot \log \sum_{j'=1}^{V} \exp(u_{j'})
\end{aligned}
\tag{1}
$$

где C – количество мультиномиальных распределений (для каждого слова в контексте), $w_{O,c}$ – c-ое слово из контекста, w_I – целевое слово (слово, подающееся на ввод модели), u – оценка принадлежности целевого слова к контексту, j_c^* - индекс определенного c-го слова из контекста в словаре [42].

Рисунок 1.2 – Иллюстрация принципа работы «окна» в алгоритме Word2Vec. Окно в данном алгоритме определяет контекст Y. X – целевое слово.

Из приведенного выше описания алгоритма можно сделать несколько выводов:

1. Контекст статичен и определяется несколькими словами, окружающими целевое слово.

2. Векторы представлений статичны. Семантика слова фиксируется «в среднем» по корпусу. Наиболее часто встречающийся контекст будет являться «решающим» для представления.

Несмотря на то, что появление данного алгоритма было прорывом в обработке естественного языка, концептуально изменив подход к кодированию языковой модели и представлению лингвистических признаков, Word2Vec являет собой способ к получению статических векторов представлений. Более современные же подходы стремятся к созданию моделей, которые позволяли бы динамически вычислять вектор представления для каждого слова (в некоторых случаях даже символа) в зависимости от контекста.

Далее будут рассмотрены современные модели, применяемые в обработке естественного языка. Стоит отметить, что все модели разработаны, руководствуясь дистрибутивной гипотезой, как и Word2Vec, поэтому исчерпывающее описание целей обучения последующих моделей было решено опустить, а вместо этого сосредоточиться на факторах, определяющих формулировку контекста, что и является предметом данного исследования.

1.3.2 Алгоритм ELMo

ELMo (Embeddings from Language Models) – модель векторных представлений, разработанная М. Питерсом [12] и представляющая собой глубокую двунаправленную языковую модель. В отличии от «традиционных» векторных представлений (статичных), представления слов ELMo являются функцией от всего ввода в модель (например, предложения). Данная модель представляет из себя рекуррентную двунаправленную LSTM нейронную сеть [19] и предназначена для обучения на задаче языкового моделирования.

Для получения зависимых от контекста представлений ELMo, в отличии от Word2Vec или GloVe, не назначает каждому слову в словаре свой вектор, а

кодирует ввод (например, предложение) посимвольно, присваивая каждому символу (которых гораздо меньше, чем слов) свой вектор представления. Затем, представления проходят через сверточный и агрегирующий слои, в которых, интуитивно, выделяются признаки, наиболее четко описывающие входную последовательность. Далее, данные проходят через двухслойную «скоростную сеть»[1] [44], где на основе весов фильтруются элементы, которые будут подаваться на вход непосредственно модели ELMo. Затем, полученные представления подаются на вход глубокой (два слоя) двунаправленной LSTM нейронной сети.

Так как задачей обучения модели является языковое моделирование, то применяется обучение без учителя. Задача сводится к предсказанию следующего слова при заданной последовательности предыдущих слов.

Для получения представлений ELMo, скрытые состояния LSTM слоев конкатенируются (прямой и обратный проходы для каждого слоя, так как сеть – двунаправленная), затем векторы (состояния) каждого слоя умножаются на веса, в зависимости от задачи и суммируются. Полученный вектор и является вектором представлением ELMo, вычисленным для отдельного слова и в зависимости от всей последовательности, в которой данное слово находится. Так как сеть рекуррентная, размер ввода может быть произвольным. Это снимает еще одно ограничение, накладываемое предыдущими моделями в виде фиксированного «окна» контекста.

1.3.3 Алгоритм BERT

BERT (Bidirectional Encoder Representations from Transformers) – модель, предназначенная для предварительного обучения глубоких двунаправленных представлений на непомеченных данных путем совместного условного совмещения левого и правого контекстов во всех слоях [8].

[1] В русской литературе не было найдено упоминаний архитектуры “Highway Network”. Вольный перевод.

На сегодняшний день BERT является моделью представления признаков, показывающей передовые результаты во многих задачах обработки естественного языка. В основе BERT лежит архитектура кодировщик-«трансформер», разработанная в качестве замены вычислительно дорогостоящим рекуррентным архитектурам.

Так же, как и ELMo, BERT имплементирует принцип трансферного обучения – для обучения некоторой модели целевой задаче возможно использование предварительно обученной модели представлений и последующая ее оптимизация для новой задачи. Авторы BERT выделяют два вида таких оптимизаций: оптимизация признаков и тонкая настройка. В то время, как ELMo использует способ оптимизации признаков – для оптимизации модели под новую задачу необходимо подобрать наиболее подходящие значения весов, влияющих на то, в каком соотношении комбинируются три слоя представлений ELMo (исходный, первый и второй слои LSTM), – BERT направлен на предварительное обучение модели с последующей возможностью тонкой настройки для конкретных задач (тонкая настройка – оптимизация всех параметров модели).

Обучение представлений BERT состоит в решении задачи Клозе, суть которой заключается в заполнении пропущенных слов в предложении, и задачи предсказания следующего предложения (например, ответа на вопрос).

В отличии от представлений ELMo, в которых двунаправленность моделируется путем обработки прямой и обратных последовательностей, представления BERT реализуют двунаправленность иначе: на этапе обучения случайные слова из входной последовательности маскируются, а задача модели – обработать контекст с обеих сторон от маскированного слова и предсказать это слово. Таким образом, реализуется явная двунаправленность, которая моделируется одновременной комбинацией двух состояний – слева и справа от целевого слова. При этом, веса, влияющие на то, в каком соотношении комбинируются контексты, являются параметрами модели и должны быть оптимизированы в процессе обучения.

Выводы по первому разделу

Из приведенных выше описаний алгоритмов и моделей видно, что существует несколько основных подходов к моделированию контекста для получения векторных представлений слов. Статичные представления (Word2Vec и GloVe) определяют контекст, как окно фиксированного размера, в котором находятся слова по обе стороны от целевого слова. Также, сами представления, полученные с использованием этих алгоритмов, предварительно обучаются один раз, а затем используются в качестве входных данных для классификаторов.

При примерно одинаковых результатах для некоторых задач, таких как извлечение именованных сущностей, сентиментальный анализ и генерирование ответов на вопросы [8, с 6-7], в основе представлений ELMo и BERT лежат разные архитектуры моделей, а как следствие – по-разному определяется контекст.

В работе выдвигается гипотеза о том, что производительность моделей обработки естественного языка зависит от реализации двунаправленности языковой модели. Ожидается, что семантика слова будет эффективнее зафиксирована моделью, в которой контекст слова определяется комбинацией левого и правого контекстов.

Для исследования была выбрана задача семантического разбора [27]. Выбор обусловлен непосредственной зависимостью значения слов от контекста и отсутствием результатов производительности моделей для данной задачи в работах, описывающих их.

Ниже приведена формализация задачи для данного исследования.

В данном исследовании задача семантического разбора представлена в виде sequence-to-sequence (последовательность к последовательности) задачи. Входное высказывание x является последовательностью слов $x_1, \dots, x_m \in V^{in}$,

где V^{in} – словарь ввода; аналогично, выходная логическая форма y является последовательностью токенов $y_1, \ldots, y_m \in V^{out}$, где V^{out} – словарь вывода.

Модели оцениваются на двух наборах данных для семантического разбора.

- **GeoQuery (Geo)** содержит вопросы на естественном языке о географии США, сопоставленные с соответствующими запросами к базе данных Prolog.
- **ATIS** содержит запросы на естественном языке к базе данных полетов, сопоставленные с соответствующими запросами к базе данных, написанными в лямбда-исчислении.

2 РАЗРАБОТКА БАЗОВОЙ МОДЕЛИ ИССЛЕДОВАНИЯ

2.1 Выбор архитектуры классификатора

Задача семантического разбора состоит в переводе текста в формальные представления в логической форме или в виде структурированных запросов. На рисунке 2.1 представлены примеры запросов (x) и их логических представлений (y), взятые из распространенных эталонных тестов (Geo, ATIS и Overnight) для задачи семантического разбора [51, 52].

GEO
x: "*what is the population of iowa ?*"
y:
```
_answer ( NV , (
  _population ( NV , V1 ) , _const (
    V0 , _stateid ( iowa ) ) ) )
```
ATIS
x: "*can you list all flights from chicago to milwaukee*"
y:
```
( _lambda $0 e ( _and
  ( _flight $0 )
  ( _from $0 chicago :  _ci )
  ( _to $0 milwaukee :  _ci ) ) )
```
Overnight
x: "*when is the weekly standup*"
y:
```
( call listValue ( call
    getProperty meeting.weekly_standup
    ( string start_time ) ) )
```

Рисунок 2.1 – Запросы на естественном языке (x) и их логические формы (y).

Многие подходы связаны с обучением моделей по предложениям, сопоставленным с их представлениями в логической форме, и следуют различным стратегиям моделирования. Примеры включают в себя использование моделей разбора [53], программирование индуктивной логики [46], вероятностные автоматы [20], классификаторы на основе строковых ядер и комбинаторные методы категориальной грамматической индукции [51, 52]. Другие работы основаны на обучении моделей семантического разбора без использования логических представлений, а, например, по предложениям,

сопоставленным с логами диалогов [54], парам вопрос-ответ [31] и дистанционном наблюдении [41].

Большинство из существующих подходов требуют наличия высококачественных лексиконов, вручную построенных шаблонов и признаков, специфичных для конкретной вариации задачи или вида логических представлений. Однако, на сегодняшний день существует обобщённый подход, который может быть легко адаптирован к различным вариациям и способам представления семантики. Использование архитектуры «кодировщик-декодеровщик», основанной на нейронных сетях, показывает хорошие результаты для задач синтактического разбора [57], машинного перевода [63], генерирования описания к изображению [62], генерировании ответа на вопрос [64] и реферирования [61].

Исходя из вышесказанного, в качестве архитектуры базовой модели для данного исследования была выбрана архитектура «кодировщик-декодировщик», а сама модель основана на модели, представленной Х. Мей [32], за исключением применения двунаправленности и многоуровневого сопоставления [34].

2.2 Общая структура базовой модели

2.2.1 Модель предсказания последовательностей

В рамках данного исследования, задача состоит в обучении модели, которая сопоставляет естественно-языковой ввод $q = x_1 \dots x_{|q|}$ с представлением его семантического значения в логической форме $a = y_1 \dots y_{|a|}$. Условную вероятность $p(a|q)$ можно представить в следующем виде:

$$p(a|q) = \prod_{t=1}^{|a|} p(y_t|y_{<t}, q) \quad (2)$$

где $y_{<t} = y_1 \dots y_{t-1}$.

Метод, выбранный для базовой модели исследования, состоит в использовании кодировщика, трансформирующего естественно-языковой ввод

q в векторное представление, и декодировщика, который обучается генерировать условную последовательность $y_1, \ldots, y_a$, зависимую от закодированного вектора.

Модель обрабатывает ввод q и вывод a как последовательности. Как показано на рисунке 2.2, кодировщик и декодировщик – две разные L-слойные рекуррентные нейронные сети с блоками долгой краткосрочной памяти (LSTM), которые рекурсивно обрабатывают токены друг за другом. Временные шаги q обрабатываются кодировщиком, а шаги a – декодировщиком. Пусть $\mathrm{h}_t^l \in \mathbb{R}^n$ обозначает состояние скрытого вектора во времени t и слое l. Тогда h_t^l вычисляется как:

$$\mathrm{h}_t^l = \mathrm{LSTM}(h_{t-1}^l, h_t^{l-1}) \tag{3}$$

где LSTM обозначает использование функции LSTM.

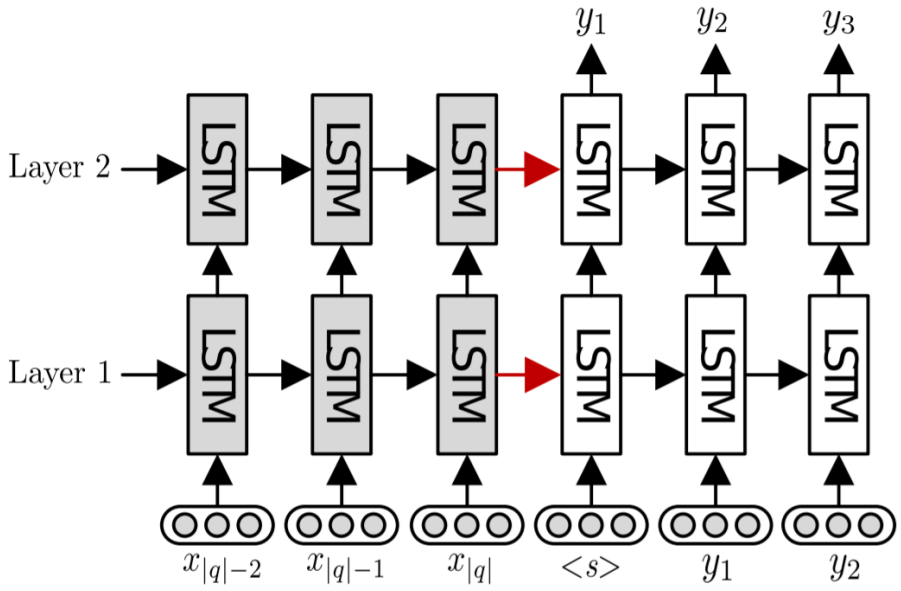

Рисунок 2.2 – Модель «кодировщик-декодировщик», состоящая из двух двухслойных рекуррентных нейронных сетей.

В кодировщике, $\mathrm{h}_t^0 = \mathbf{W}_q \mathbf{e}(x_t)$ – векторное представление слова для текущего токена из ввода, при $\mathbf{W}_q \in \mathbb{R}^{n \times V_q}$, являющейся параметрической матрицей, а $\mathbf{e} \cdot$ – индексом соответствующего токена.

В декодировщике, $\mathrm{h}_t^0 = \mathbf{W}_q \mathbf{e}(y_{t-1})$ – векторное представление слова предыдущего предсказанного слова, где $\mathbf{W}_q \in \mathbb{R}^{n \times V_a}$. Кодировщик и декодировщик имеют разные LSTM параметры.

После того, как токены входной последовательности $x_1, \ldots, x_q$ закодированы в векторы, они используются для инициализации скрытых состояний первого временного шага в кодировщике. Затем, скрытый вектор последнего слоя LSTM h_t^L в декодировщике используется для предсказания t-го выходного токена как:

$$\mathrm{p}\ y_t\ y_{<t}, q\ = \mathrm{softmax}\ \mathbf{W}_o \mathbf{h}_t^L\ ^{\mathrm{T}} \mathbf{e}(y_t) \qquad (4)$$

где $\mathbf{W}_o \in \mathbb{R}^{V_a \times n}$ – параметрическая матрица, а $\mathbf{e}(y_t) \in 0{,}1^{V_a}$ – унитарный вектор для вычисления вероятности y_t из предсказанного распределения.

Каждая последовательность дополняется токенами «начало последовательности» – <s> и «конец последовательности» – </s>. Процесс генерирования останавливается, когда предсказан токен </s>. Затем, условная вероятность генерирования полной последовательности $p(a|q)$ рассчитывается с помощью уравнения (2).

2.2.2 Механизм внимания

Как показано в уравнении (4), скрытые векторы входной последовательности не используются напрямую в процессе декодировки. Интуитивно, во входных данных только некоторая часть является релевантной для предсказывания выходного токена. В исследовании Д. Бахданау были предложены различные методы для интегрирования информации кодировщика (в форме контекстного вектора) в каждый временной шаг декодировщика [55].

Для поиска релевантного контекста в информации кодировщика пр и текущем скрытом состоянии $\mathbf{h}_t^L$ декодировщика, вычисляется его индекс внимания с использованием k-го скрытого состояния кодировщика как:

$$\mathrm{s}_k^t = \frac{\exp\{\mathbf{h}_k^L \cdot \mathbf{h}_t^L\}}{\sum_{j=1}^{q} \exp\{\mathbf{h}_j^L \cdot \mathbf{h}_t^L\}} \tag{5}$$

где $\mathbf{h}_1^L, \ldots, \mathbf{h}_q^L$ – скрытые векторы верхнего слоя кодировщика. Тогда контекстный вектор является взвешенной суммой скрытых векторов кодировщика:

$$\mathbf{c}^t = \sum_{k=1}^{q} \mathrm{s}_k^t \mathbf{h}_k^L \tag{6}$$

Вместо уравнения (4), этот контекстный вектор, который служит «резюме» кодировщика, далее используется для вычисления вероятности генерирования y^t как:

$$\mathbf{h}^{att} = \tanh\ \mathbf{W}_1 \mathbf{h}_t^L + \mathbf{W}_2 \mathbf{c}^t \tag{7}$$

$$\mathrm{p}\ y_t\ y_{<t}, q\ = \mathrm{softmax}\ \mathbf{W}_o \mathbf{h}_t^{att}\ ^{\mathrm{T}} \mathbf{e}(y_t) \tag{8}$$

где $\mathbf{W}_o \in \mathbb{R}^{V_a \times n}$ и $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{n \times n}$ – три параметрические матрицы, а $\mathbf{e}(y_t)$ – унитарный вектор, используемый для расчета вероятности y_t.

2.2.3 Обучение модели

Цель обучения – максимизировать вероятность генерирования логических форм для соответствующих запросов на естественном языке. Таким образом, целевая функция модели:

$$\text{minimize} - \sum_{q,a \in D} \log p(a|q) \tag{9}$$

где D – множество всех пар «естественный язык-логическая форма»; $p(a|q)$ вычисляется как показано в уравнении (2).

Для решения данной невыпуклой задачи оптимизации применяется алгоритм RMSProp. Также, для регуляризации модели, применяется метод исключения (отключения случайных нейронов) [50]. В частности, операторы исключения применяются между слоями LSTM и после скрытых слоев перед softmax классификаторами. Данная техника позволяет в значительной мере

решить проблему переобучения, особенно для наборов данных небольших размеров.

2.3 Результаты моделирования

2.3.1 Программная реализация

Для программной реализации модели и проведения вычислительных экспериментов был выбран язык Python и библиотека машинного обучения Tensorflow.

Программное решение состоит из четырех модулей:

- accuracy;
- data_utils;
- parse_s2s_att;
- seq2seq_model.

Модуль parse_s2s_att отвечает за оркестрацию процесса обучения и тестирования модели. Модуль запускается CLI командой и на основе переданных параметров выполняет одну из четырех функций: обучение, интерактивная декодировка, «самотестирование» модели на случайных данных (для отладки архитектуры) и оценка модели на тестовых данных. На рисунке 2.3 представлена листинг функции main.

```
def main(_):
    if FLAGS.self_test:
        self_test()
    elif FLAGS.decode:
        decode()
    elif FLAGS.test:
        from_test_data = os.path.join(FLAGS.data_dir, "test_q.txt")
        to_test_data = os.path.join(FLAGS.data_dir, "test_f.txt")
        if not (os.path.exists(from_test_data) and os.path.exists(to_test_data)):
            if FLAGS.test_file:
                data_utils.splitToFrom(FLAGS.data_dir, FLAGS.test_file, "test")  # split to-from
            else:
                print("test data file missing!")
                return

        print("computing test accuracy...")
        test_acc = test_accuracy(from_test_data, to_test_data)
        print("test accuracy =", test_acc)
    else:
        train()
```

Рисунок 2.3 – Листинг функции main модуля parse_s2s_att.

Модуль data_utils содержит функции для предварительной обработки данных, такие как токенизация, построение словаря, разделение набора данных и т.д.

В модуле seq2seq_model определяется класс модели. На рисунке 2.4 представлена диаграмма класса Seq2SeqModel.

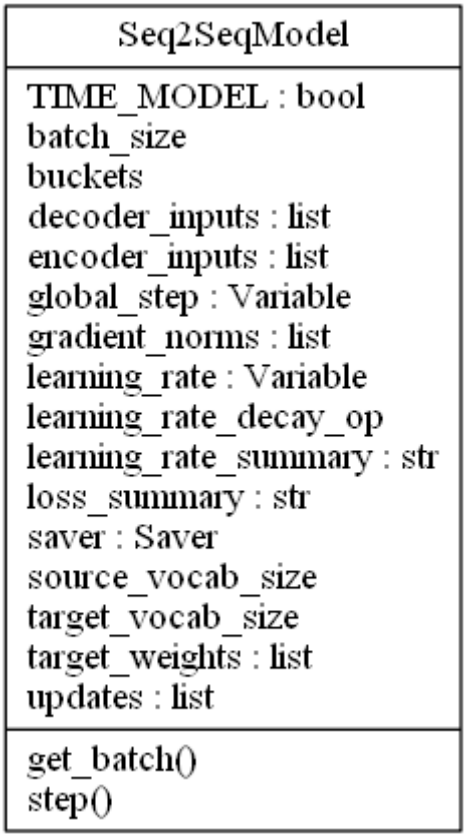

Рисунок 2.4 – Диаграмма класса Seq2SeqModel.

В конструкторе класса Seq2SeqModel выполняется построение вычислительного графа Tensorflow. Класс реализует два метода: получение партии данных для шага обучения и выполнение одного шага обучения. Предварительная обработка данных и оркестрация всего процесса обучения (запуск и конфигурация сессии Tensorflow, итерирование по данным обучения и т.д.) производится в модуле parse_s2s_att.

Модуль accuracy содержит функции для вычисления точности разбора. Точность разбора вычисляется как пропорция верно разобранных предложений из всего набора данных.

2.3.2 Параметры модели

Входные предложения естественного языка были приведены к нижнему регистру. Слова были обработаны алгоритмом стемминга NLTK. Слова, встречающиеся в наборах данных только один раз, были отфильтрованы (только для входных предложений).

Для оптимизации применялся алгоритм RMSProp (с размером партии в 20 образцов). Значение константы сглаживания алгоритма RMSProp было установлено в 0,95.

Параметры модели были случайно инициализированы из равномерного распределения $U(-0.08, 0.08)$. Кодировщик и декодировщик содержали один слой LSTM. Размерность скрытых векторов и представлений слов была установлена в 100 измерений. Частота отключения случайных нейронов – 0,3.

В целях оценки производительности модели, представления слов вычислялись в процессе обучения (наравне со всеми параметрами модели), а не являлись предварительно обученными. Такой подход эффективен только для конкретной задачи и конкретного набора данных, так как количество полученных представлений ограничено количеством уникальных слов в наборе данных.

Количество итераций обучения регулировалось методом ранней остановки с параметром терпимости равным 30 итерациям (прекращение обучения, если на протяжении 30 итераций значение ошибки для множества данных тестирования не уменьшается).

Входные предложения были инвертированы. Для генерирования логических форм применяется алгоритм жадного выбора.

2.3.3 Анализ результатов базовой модели исследования

Результаты моделирования базовой модели оценивались на наборах данных Geo и ATIS. В таблицах 1 и 2 приведены результаты базовой модели в сравнении с другими подходами к решению данных задач. Для методов SCISSOR, WASP и LNLZ08 применялась перекрестная проверка с коэффициентом $k = 10$. Для ZC05, ZC07, KCAZ13 и TISP применялось стандартное разделение дынных на наборы для тестирования и обучения, как описано в имплементации ZC05.

Таблица 1 – Результаты для набора данных Geo. Базовая модель исследования – Seq2Seq.

Метод	**Точность**
SCISSOR [16]	72.3
WASP [27]	74.8
LNLZ08 [4]	81.8
ZC05 [51]	79.3
ZC07 [52]	86.1
TISP [53]	88.9
Seq2Seq	84.3

Таблица 2 – Результаты для набора данных ATIS.

Метод	**Точность**
ZC07	84.6
FUBL [59]	82.8
GUSP++ [60]	83.5
TISP	88.9
Seq2Seq	84.1

Точность моделей определялась как пропорция верно разобранных входных предложений на их соответствующие логические формы.

Исходя из полученных результатов, можно сделать вывод о том, что разработанная модель может быть использована для решения задачи семантического разбора. Архитектура кодировщика-декодировщика занимает четвертое и третье место для приведенных в сравнение задач.

Одним из факторов, ограничивающих производительность данной модели для набора данных Geo, скорее всего, является древоподобная структура логических форм в данном наборе, в то время как модель производит вывод в виде линейной последовательности. Тем не менее, модель показывает достаточные результаты для возможности проведения дальнейших экспериментов.

Выводы по второму разделу

Во втором разделе были рассмотрены существующие методы решения задачи семантического разбора. В ходе анализа источников, было выявлено, что наиболее обобщенным и простым подходом является использование модели sequence-to-sequence, основанной на нейронных сетях. Применение такой модели также позволяет в дальнейшем оценить производительность моделей представлений слов, так как для их внедрения необходимо лишь заменить словарь и обучаемый слой входных представлений на предварительно обученную модель представлений слов.

Для дальнейшего проведения экспериментов с моделями представлений была разработана модель классификатора на основе рекуррентной нейронной сети с архитектурой «кодировщик-декодировщик». В результате проведения серии экспериментов было выявлено, что данная архитектура пригодна для решения задачи семантического разбора.

3 АНАЛИЗ И ИНТЕРПРЕТАЦИЯ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

3.1 Проведение вычислительных экспериментов

3.1.1 Построение экспериментальных моделей

Для проверки гипотезы о том, что производительность моделей обработки естественного языка зависит от реализации двунаправленности в этих моделях, была проведена серия экспериментов с различными моделями представлений слов. Для проведения таких экспериментов, базовая модель, описанная во втором разделе, была изменена следующим образом:

- слой представлений, в котором входные токены трансформировались в векторные представления (обучаемые вместе с другими параметрами модели), был заменен на модели предварительно обученных векторных представлений (ELMo и BERT). Также, в целях сравнения производительности статичных представлений слов и вычисляемых в зависимости от контекста, был проведен эксперимент с представлениями слов GloVe.
- Регуляризация модели путем случайного отключения нейронов не применялась (для оценки того, насколько использование предварительно обученных представлений помогает в регуляризации модели).

Предварительно обученные представления используются только для кодирования входных в кодировщик запросов на английском языке. Слой представлений декодировщика так же, как и в базовой модели, инициализируется случайным образом и обучается с другими параметрами сети.

Векторы представлений GloVe оптимизируются во время обучения модели. Для моделей представления BERT и ELMo параметры фиксируются, но вместо этого для них обучаются «скалярные комбинации» активаций отдельных слоев. Таким образом, ввод к LSTM кодировщику для $\boldsymbol{k}$-го токена $\boldsymbol{t}_k$ в последовательности выглядит следующим образом:

$$\mathbf{e}_k = [\mathbf{r}_k, \mathbf{g}_k] \tag{10}$$

где $\boldsymbol{g}_k$ – 100-мерное представление GloVe токена $\boldsymbol{t}_k$, а $\boldsymbol{r}_k$ – предварительно обученное представление $\boldsymbol{t}_k$:

$$\mathbf{r}_k = \gamma \sum_{j=0}^{L} \mathbf{s}_j \mathbf{h}_{k,j} \tag{11}$$

где $\mathbf{h}_{k,j}$ – активация, ассоциированная с k-тым токеном в j-ом слое предварительно обученной модели, γ – обученный скалярный параметр, а $\mathbf{s} = \mathrm{softmax}(\mathbf{w})$, где $\mathbf{w}$ – L-мерный обучаемый вес.

Такой подход «скалярной комбинации», по сути, эквивалентен подходу, применяемому в ELMo, в то время, как BERT использует подход тонкой настройки, когда вся языковая модель настраивается (обучается) для конкретной задачи. Для использования подхода скалярного комбинирования в экспериментах данного исследования есть несколько мотиваций. Во-первых, он решает вопрос необходимости выбора наиболее подходящего подхода к тонкой настройке. Не существует консенсуса по наилучшему методу тонкой настройки, позволяющему полностью избежать проблемы «катастрофического забывания», когда знание, хранящееся в предварительно обученной модели, разрушается в процессе оптимизации весов модели для конкретной задачи. BERT не использует техник, предотвращающих эту проблему. Поэтому, фиксируя языковые модели и выучивая то, как наилучшим образом использовать информацию уже находящуюся в них, возможно наиболее объективно сравнить их эффективность.

Существует две версии BERT: BERT (Base) и BERT (Large). BERT (Base) подразумевается, как практически идентичная модель его предшественнику – OpenAI GPT [23]. Как и OpenAI GPT, модель BERT (Base) является 12-слойным трансформером и генерирует 768-мерные представления, которые превращаются в 868-мерные векторы ввода в кодировщик после конкатенации с представлениями GloVe. BERT (Large) является 24-слойным трансформером, производящим 1024-мерные представления, которые превращаются в 1124-мерный ввод LSTM кодировщика.

3.1.2 Конфигурация экспериментов

Для объективного сравнения производительности различных предварительно обученных представлений, все эксперименты разделяли практически одинаковые конфигурации, за исключением, очевидно, модели представлений. Обучение всех моделей проходило на одном графическом процессоре NVIDIA GeForce 960M с индексом Compute Capability – 5,0. В качестве алгоритма оптимизации применялся RMSProp. Во всех экспериментах использовался размер партий 16 и шаг обучения 0,01.

Остальные параметры моделей и разделение набора данных неизменны и представлены в подразделе 2.3.

3.2 Анализ полученных результатов

В таблице 3 приведены результаты для каждой модели и двух наборов данных.

Таблица 3 – Результаты производительности для четырех моделей исследования. Точность замерена на уровне последовательностей.

Модель	**ATIS**	**Geo**
Seq2Seq	79.7	68.9
Seq2Seq & ELMo	83.2	**75.7**
Seq2Seq & Bert (Base)	**83.4**	75.6
Seq2Seq & Bert (Large)	82.9	73.1

В таблицах 4 и 5 представлены подробные результаты для каждого набора данных.

Таблица 4 – Подробные результаты экспериментов на наборе данных ATIS.

Модель	**Тестовая точность**	**Точность валидации**	**Лучшая итерация**	**Время обучения (м:с)**	**Время на итерацию (с)**
Seq2Seq	79.7	83.1	79	**60:18**	**33.15**
Seq2Seq & ELMo	83.2	**86.9**	105	140:14	62.24
Seq2Seq & Bert (Base)	**83.4**	86.1	**71**	105:56	61.27
Seq2Seq & Bert (Large)	82.9	85.8	74	141:18	82.19

Таблица 5 – Подробные результаты экспериментов на наборе данных Geo.

Модель	**Тестовая точность**	**Точность валидации**	**Лучшая итерация**	**Время обучения (м:с)**	**Время на итерацию (с)**
Seq2Seq	68.9	70.6	103	**8:59**	**4.02**
Seq2Seq & ELMo	**75.7**	77.2	81	15:60	8.22
Seq2Seq & Bert (Base)	75.6	**78.9**	71	12:26	7.38
Seq2Seq & Bert (Large)	73.1	74.7	**47**	16:08	12.58

Применение предварительно обученных представлений явно улучшает производительность базовой модели (Seq2Seq) для каждого набора данных. Разница в абсолютной точности между базовой моделью и следующей худшей, в которой применяются предварительно обученные представления, составляет 3.2% и 4.2% для ATIS и Geo соответственно, в то время, как разница между базовой и самой производительной моделями составляет 3.7% и 6.8% соответственно. С уменьшением набора данных, разница в производительности над базовой моделью растет, также, как растет разница между наихудшей и наилучшей моделями с предварительно обученными моделями представлений. Этот факт подтверждает часто встречающееся в литературе высказывание: предварительное обучение наиболее эффективно влияет на производительность для небольших наборов данных.

Результаты показывают, что включение признаков, которые были предварительно обучены в качестве языковой модели, в модель кодировщика-декодировщика является простым способом улучшения производительности.

Гораздо менее очевидно, какая из трех предварительно обученных моделей наиболее эффективна. Это, что занимательно, не BERT (Large) – модель, которая показывает наилучшие результаты для многих задач обработки естественного языка (однако, в оригинальной публикации данной модели отсутствуют результаты для задачи семантического разбора), – показавшая даже худшие результаты в сравнении с ELMo, а также являющаяся самой вычислительно дорогой, в плане времени, затрачиваемого на итерацию. Очевидно, что это не связано с архитектурой модели или набором данных, на котором данная модель была обучена, так как эти факторы присущи и BERT (Base). Вероятно, что фактором, оказывающим влияние в этой ситуации, является увеличенная размерность модели, приводящая к переобучению. Возможно, регуляризация каждой модели может привести BERT (Large) к наилучшим показателям точности среди других моделей, но сама по себе данная модель обладает наихудшим регуляризирующим эффектом.

Что касается производительности ELMo и BERT (Base), результаты для двух тестов отличаются. Это может быть обусловлено тем, что ATIS содержит гораздо больше экземпляров данных (5410 запросов), чем Geo (880 запросов). Высокий показатель точности модели BERT (Base) для ATIS, вероятнее всего, обусловлен тем, как реализована двунаправленность этой модели. В то время, как BERT (Base) использует словарь WordPiece (основанный на разбиении слов по слогам), в ELMo реализован символьный подход к изначальному кодированию представлений. Таким образом, благодаря меньшим размерам Geo, модель с представлениями ELMo смогла показать большую точность, так как не имеет проблем, связанных с отсутствием слов в словаре. Однако, на большем количестве данных, BERT (Base), вероятно, также гораздо реже сталкивается с этой проблемой. Это позволяет сделать вывод о том, что при прочих равных, модель с представлениями BERT (Base) является наиболее эффективной среди остальных.

Таким образом, результаты экспериментов подтвердили гипотезу о том, что производительность моделей обработки естественного языка зависит от реализации двунаправленности в моделях представлений, а именно: наиболее эффективным способ реализации двунаправленности является объединение левого и правого контекстов целевого слова.

Выводы по третьему разделу

На основе рекуррентного классификатора с архитектурой «кодировщик-декодировщик» были проведены эксперименты с тремя моделями представлений: GloVe, ELMo и BERT (в двух модификациях). Анализ результатов, полученных в ходе экспериментов, позволил выявить следующие факты:

1. Двунаправленность контекста, реализованная путем объединения левого и правого контекстов слова, является более эффективным подходом, по сравнению с рекуррентным, к моделированию семантических характеристик слова.

2. Использование представлений с большим количеством измерений для задач с относительно небольшими объемами данных приводит к худшей регуляризации модели.

ЗАКЛЮЧЕНИЕ

Одной из главных проблем обработки естественного языка является подход к представлению текстовых данных в виде, пригодном для дальнейшей работы с ними в математических моделях. В то время, как плотные векторные представления стали стандартом для кодирования слов, начиная с 2013 года [15], сегодня существуют и развиваются различные подходы к моделированию контекста в этих представлениях.

Выводы проведенного исследования:

1. Подход к моделированию признаков является основополагающим фактором, влияющим на производительность моделей обработки естественного языка. Так, в ходе исследования были определены современные методы представлений слов, заключающиеся в динамическом моделировании векторных представлений, зависимых от контекста.

2. Модель, построенная на основе нейронных сетей, с архитектурой «кодировщик-декодировщик» может успешно применяться для решения задачи генерирования последовательностей, разновидностью которой является задача семантического разбора. В ходе исследования были получены результаты, доказывающие конкурентоспособность таких моделей в сравнении с ранее применяемыми решениями данной задачи.

3. Сравнительный анализ производительности предварительно обученных языковых моделей (ELMo и BERT) показал, что подход к реализации двунаправленности является важным фактором в моделировании представлений слов. Так, рекуррентный подход ELMo оказался менее эффективным, уступая подходу, реализованному в архитектуре «трансформер». Условная комбинация левого и правого контекстов целевого слова позволяет лучше зафиксировать его семантическое значение.

4. Использование предварительно обученных представлений не приносит достаточного эффекта регуляризации модели, когда размерность модели представлений значительно выше размерности задачи. Использование модели представлений BERT (Large) с количеством измерений в 1024 привело к наихудшим результатам среди всех проведенных экспериментов. Данный факт открывает поле для дальнейшей работы по исследованию оптимальных методов регуляризации при использовании предварительно обученных языковых моделей.

Помимо достигнутой изначальной цели исследования, были также получены новые данные о влиянии использования предварительно обученных представлений на регуляризацию моделей. Полученные результаты могут быть использованы при выборе подхода к разработке новых языковых моделей или при выборе существующей модели представлений для решения задач обработки естественного языка.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Кудинов, М.С. О применимости рекуррентных нейронных сетей к задаче статистического моделирования русского языка / М. С. Кудинов // Журнал сибирского федерального университета. Серия: техника и технологии. – 2016. – С. 1291-1301.

2. Соловьева, Е.Б. Классы рекуррентных нейронных сетей для моделирования нелинейных динамических систем / Е. Б. Соловьева // Международная конференция по мягким вычислениям и измерениям. – 2017. – С. 159-162.

Электронные ресурсы

3. A Dependency-Based Neural Network for Relation Classification [Электронный ресурс] / Y. Liu, F. Wei, S. Li и др. – 2015. – Режим доступа: https://arxiv.org/pdf/1507.04646.

4. A Generative Model for Parsing Natural Language to Meaning Representations [Электронный ресурс] / W. Lu, H.T Ng, W.S. Lee, L.S. Zettlemoyer // Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. – 2008. – С. 786-792. – Режим доступа: https://www.aclweb.org/anthology/D08-1082.

5. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses [Электронный ресурс] / A. Sordoni, M. Galley, M. Auli и др. – 2015. – Режим доступа: https://arxiv.org/pdf/1506.06714.

6. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification [Электронный ресурс] / L. Dong, F. Wei, C. Tan и др. // Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers). – 2014. – С. 49-54. – Режим доступа: https://www.aclweb.org/anthology/P14-2009.

7. Attention Is All You Need [Электронный ресурс] / A. Vaswani, N. Shazeer, N. Parmar и др. – 2017. – Режим доступа: https://arxiv.org/pdf/1706.03762.

8. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Электронный ресурс] / D Jacob, C. Ming-Wei, L. Kenton, T. Kristina. – 2018. – Режим доступа: https://arxiv.org/pdf/1810.04805v2.

9. Bitvai, Z. Non-Linear Text Regression with a Deep Convolutional Neural Network [Электронный ресурс] / Z. Bitvai, T. Cohn // Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers). – 2015. – С. 180-185. – Режим доступа: https://www.aclweb.org/anthology/P15-2030.

10. Chrupała, G. Normalizing tweets with edit scripts and recurrent neural embeddings [Электронный ресурс] / G. Chrupała // Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers). – 2014. – С. 680-686. – Режим доступа: https://pdfs.semanticscholar.org/351d/d41349bcc86331d 9d602970df9e4159b72e9.pdf.

11. Decoding with Large-Scale Neural Language Models Improves Translation [Электронный ресурс] / A. Vaswani, Y. Zhao, V. Fossum, D. Chiang // Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. – 2013. – С. 1387-1392. – Режим доступа: https://www.aclweb.org/anthology/D13-1140.

12. Deep contextualized word representations [Электронный ресурс] / M.E. Peters, M. Neumann, M. Iyyer и др. – 2018. – Режим доступа: https://arxiv.org/pdf/1802.05365.

13. Distributed Representations of Words and Phrases and their Compositionality [Электронный ресурс] / T. Mikolov, I. Sutskever, K. Chen и др. – 2013. – Режим доступа: https://arxiv.org/pdf/1310.4546.pdf.

14. Durret, G. Neural CRF Parsing [Электронный ресурс] / G. Durret, D. Klein. – 2015. – Режим доступа: https://arxiv.org/pdf/1507.03641.

15. Efficient Estimation of Word Representations in Vector Space [Электронный ресурс] / T. Mikolov, K. Chen, G. Corrado, J. Dean // Proceedings of

the International Conference on Learning Representations (ICLR). – 2013. – Режим доступа: https://arxiv.org/pdf/1301.3781.pdf.

16. Ge R. A statistical semantic parser that integrates syntax and semantics [Электронный ресурс] / R. Ge, R.J. Mooney // Proceedings of CoNLL. – 2005. – С. 9-16. – Режим доступа: http://www.cs.utexas.edu/~ml/papers/parsing-conll-05.pdf.

17. Gispet, A. Fast and Accurate Preordering for SMT using Neural Networks [Электронный ресурс] / A. Gispet, G. Iglesias, B. Byrne // Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL. – 2015. – С. 1012-1017. – Режим доступа: https://pdfs.semanticscholar.org/e2cc/4e5c70070dd5914cfb65224b674de31875f6.pdf

18. Glorot, X. Understanding the difficulty of training deep feedforward neural networks [Электронный ресурс] / X. Glorot, Y. Bengio. – 2010. – Режим доступа: http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf.

19. Graves, A. Generating Sequences With Recurrent Neural Networks [Электронный ресурс] / A. Graves. – 2013. – Режим доступа: https://arxiv.org/pdf/1308.0850.

20. He, Y. Semantic processing using the Hidden Vector State model [Электронный ресурс] / Y. He, S. Young. – 2003. – Режим доступа: https://www.sciencedirect.com/science/article/pii/S0885230804000117?via%3Dihub.

21. Henderson, M. Deep Neural Network Approach for the Dialog State Tracking Challenge [Электронный ресурс] / M. Henderson, B. Thomson, S. Young // Proceedings of the SIGDIAL 2013 Conference. – 2013. – С. 467-471. – Режим доступа: https://aclweb.org/anthology/W13-4073.

22. Hermann, K.M. Multilingual Distributed Representations without Word Alignment [Электронный ресурс] / K.M. Hermann, P. Blunsom. – 2013. – Режим доступа: https://arxiv.org/pdf/1312.6173.

23. Improving Language Understanding by Generative Pre-Training [Электронный ресурс] / A. Radford, K. Narasimhan, T. Salimans, I. Sutskever. – 2018. – Режим доступа: https://openai.com/blog/language-unsupervised/.

24. Isroy, O. Deep recursive neural networks for compositionality in language [Электронный ресурс] / O. Isroy, C. Cardie // NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems. – Vol. 2. – 2014. – С. 2096-2104. – Режим доступа: https://dl.acm.org/citation.cfm?id=2969061.

25. Jia, R. Data Recombination for Neural Semantic Parsing [Электронный ресурс] / R. Jia, P. Liang. – 2016. – Режим доступа: https://arxiv.org/pdf/1606.03622.

26. Johnson, R. Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding [Электронный ресурс] / R. Johnson, T. Zhang. – 2015. – Режим доступа: https://arxiv.org/pdf/1504.01255.

27. Kate, R.J. Using String-Kernels for Learning Semantic Parsers [Электронный ресурс] / R.J. Kate, R.J. Mooney // Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. – 2006. – С. 913-920. – Режим доступа: https://www.aclweb.org/anthology/P06-1115.

28. Kim, Y. Convolutional Neural Networks for Sentence Classification [Электронный ресурс] / Y. Kim. – 2014. – Режим доступа: https://arxiv.org/pdf/1408.5882.

29. Large Language Models in Machine Translation [Электронный ресурс] / T. Brants, A.C. Popat, P. Xu и др. // Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. – 2007. – С. 858-867. – Режим доступа: https://www.aclweb.org/anthology/D07-1090.pdf.

30. Li, J. A Model of Coherence Based on Distributed Sentence Representation [Электронный ресурс] / J. Li, E. Hovy // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). – 2014. – С. 2039-2048. – Режим доступа: http://www.cs.cmu.edu/~./hovy/papers/14EMNLP-discourse-coherence.pdf.

31. Liang, P. Learning Dependency-Based Compositional Semantics [Электронный ресурс] / P. Liang, M.I. Jordan, D. Klein. – 2013. – Режим доступа: https://www.aclweb.org/anthology/J13-2005.

32. Mei, H. Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences [Электронный ресурс] / H. Mei, M. Bansal, M.R. Walter. – 2016. – Режим доступа: https://arxiv.org/pdf/1506.04089.

33. Multi-domain Dialog State Tracking using Recurrent Neural Networks [Электронный ресурс] / N. Mrkšić, D. Ó Séaghdha, B. Thomson и др. – 2015. – Режим доступа: https://arxiv.org/pdf/1506.07190.

34. Multi-Level Alignments As An Extensible Representation Basis for Textual Entailment Algorithms [Электронный ресурс] / T. Noh, S. Pado, V. Shwartz и др. – 2015. – Режим доступа: https://www.aclweb.org/anthology/S15-1022.

35. Natural Language Processing (almost) from Scratch [Электронный ресурс] / R. Collobert, J. Weston, L. Bottou и др. – 2011. – Режим доступа: https://arxiv.org/pdf/1103.0398.

36. Nguyen, T.H. Relation Extraction: Perspective from Convolutional Neural Networks [Электронный ресурс] / T.H. Nguyen, R. Grishman // Proceedings of NAACL-HLT 2015. – 2015. – С. 39-48. – Режим доступа: https://www.aclweb.org/anthology/W15-1506.

37. Parsing with Compositional Vector Grammars [Электронный ресурс] / R. Socher, J. Bauer, C.D. Manning, A.Y. Ng // Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. – 2013. – С. 455-465. – Режим доступа: https://www.aclweb.org/anthology/P13-1045.

38. Pennington, J. GloVe: Global Vectors for Word Representation [Электронный ресурс] / J. Pennington, R. Socher, C.D. Manning. – 2014. – Режим доступа: https://nlp.stanford.edu/pubs/glove.pdf.

39. Question Answering over Freebase with Multi-Column Convolutional Neural Networks [Электронный ресурс] / L. Dong, F. Wei, M. Zhou, K. Xu // Proceedings of the 53rd Annual Meeting of the Association for Computational

Linguistics and the 7th International Joint Conference on Natural Language Processing. – 2015. – С. 260-269. – Режим доступа: https://www.aclweb.org/anthology/P15-1026.

40. Recurrent neural network based language model [Электронный ресурс] / T. Mikolov, M. Karafiat, L. Burget и др. – 2010. – Режим доступа: https://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf.

41. Reddy, S. Large-scale Semantic Parsing without Question-Answer Pairs [Электронный ресурс] / S. Reddy, M. Lapata, M. Steedman // Transactions of the Association for Computational Linguistics. – 2014. – С. 377-392. – Режим доступа: https://www.aclweb.org/anthology/Q14-1030.

42. Rong, X. word2vec Parameter Learning Explained [Электронный ресурс] / X. Rong. – 2014. – Режим доступа: https://arxiv.org/pdf/1411.2738.

43. Rumelhart, D.E. A general framework for parallel distributed processing [Электронный ресурс] / D.E. Rumelhart, G.E. Hinton, J.L. McClelland // Parallel distributed processing: explorations in the microstructure of cognition. – Cambridge, 1986. – С. 45-76. – Режим доступа: https://dl.acm.org/citation.cfm?id=104286.

44. Srivasatva, R.K. Highway Networks [Электронный ресурс] / R.K. Srivasatva, K. Greff, J. Schmidhuber. – 2015. – Режим доступа: https://arxiv.org/pdf/1505.00387.

45. Structured Training for Neural Network Transition-Based Parsing [Электронный ресурс] / D. Weiss, C. Alberti, M. Collins, S. Petrov. – 2015. – Режим доступа: https://arxiv.org/pdf/1506.06158.

46. Thompson, C. Acquiring Word-Meaning Mappings for Natural Language Interfaces [Электронный ресурс] / C. Thompson, R.J. Mooney. – 2003. – Режим доступа: https://arxiv.org/pdf/1106.4571.

47. Transition-Based Dependency Parsing with Stack Long Short-Term Memory [Электронный ресурс] / C. Dyer, M. Ballesteros, W. Ling и др. – 2015. – Режим доступа: https://arxiv.org/pdf/1505.08075.

48. Translation Modeling with Bidirectional Recurrent Neural Networks [Электронный ресурс] / M. Sundermeyer, T. Alkhouli, J. Wuebker, H. Ney // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). – 2014. – С. 14-25. – Режим доступа: https://www.aclweb.org/anthology/D14-1003.

49. Yin, W. Multichannel Variable-Size Convolution for Sentence Classification [Электронный ресурс] / W. Yin, H. Schutze // Proceedings of the 19th Conference on Computational Language Learning. -2015. – С. 204-214. – Режим доступа: https://aclweb.org/anthology/K15-1021.

50. Zaremba, W. Recurrent Neural Network Regularization [Электронный ресурс] / W. Zaremba, I. Sutskever, O. Vinyals. – 2015. – Режим доступа: https://arxiv.org/pdf/1409.2329.

51. Zettlemoyer, L.S. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars [Электронный ресурс] / L.S. Zettlemoyer, M. Collins. – 2005. – Режим доступа: https://homes.cs.washington.edu/~lsz/papers/zc-uai05.pdf.

52. Zettlemoyer, L.S. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form [Электронный ресурс] / L.S. Zettlemoyer, M. Collins. – 2007. – Режим доступа: https://homes.cs.washington.edu/~lsz/papers/zc-emnlp07.pdf.

53. Zhao, K. Type-Driven Incremental Semantic Parsing with Polymorphism [Электронный ресурс] / K. Zhao, L. Huang. – 2015. – Режим доступа: https://arxiv.org/pdf/1411.5379.

54. Artzi, Y. Bootstrapping semantic parsers from conversations [Текст] / Y. Artzi, L. Zettlemoyer // Proceedings of the 2011 EMNLP. – Edinburgh, 2011. – С. 421-432.

55. Bahdanau, D. Neural machine translation by jointly learning to align and translate [Текст] / D. Bahdanau, K. Cho, Y. Bengio // Proceedings of the ICLR. – San Diego, 2015.

56. Elman, Jeffrey L. Finding Structure in Time [Текст] / Jeffrey L. Elman // COGNITIVE SCIENCE 14. – San Diego: University of California. – 1990. – С. 179-211.

57. Grammar as a foreign language [Текст] / O. Vinyals, L. Kaiser, T. Koo и др. // Advances in Neural Information Processing Systems 28. – 2015. –С. 2755-2763.

58. Küchler, A. Inductive learning in symbolic domains using structure-driven recurrent neural networks [Текст] / A. Küchler, C. Goller // Lecture Notes in Computer Science (1137). – 1996. – С. 183-197.

59. Lexical generalization in CCG grammar induction for semantic parsing [Текст] / T. Kwiatkowski, L. Zettlemoyer, S. Goldwater, M. Steedman // Proceedings of the 2011 EMNLP. – Edinburgh, 2011. – С. 1512-1523.

60. Poon, H. Grounded unsupervised semantic parsing [Текст] / H. Poon // Proceedings of the 51st ACL. – Sofia, 2013. – С. 933-943.

61. Rush, A.M. A neural attention model for abstractive sentence summarization [Текст] / A.M. Rush, S. Chopra, J. Weston // Proceedings of the 2015 EMNLP. – Lisbon, 2015. – С. 379-389.

62. Show and tell: A neural image caption generator [Текст] / O. Vinyals, A. Toshev, S. Bengio, D. Erhan // Proceedings of CVPR. – Boston, 2015. –С. 3156-3164.

63. Teaching machines to read and comprehend [Текст] / K.M. Hermann, T. Kocizsky, E. Grefenstette и др. // Advances in Neural Information Processing Systems 28. – 2015. – С. 1684-1692.

64. Sutskever, I. Sequence to sequence learning with neural network [Текст] / I. Sutskever, O. Vinyals, Q.V. Lee // Advances in Neural Information Processing Systems 27. – 2014. – С. 3104-3112.