Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ВЫСШАЯ ШКОЛА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Направление: 09.03.03 Прикладная информатика

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

КЛАССИФИКАЦИЯ ТОНАЛЬНОСТИ ТЕКСТОВ МЕТОДАМИ ГЛУБОКОГО ОБУЧЕНИЯ

Студент 4 курса

группы 11-504

«___»_____2019 г. Мустафин М.Р.

К. Ф.-м. н., ассистент

кафедры интеллектуальных

технологий поиска

«___»_____2019 г. Тутубалина Е.В.

Директор Высшей школы ИТИС

«___»_____2019 г. Хасьянов А.Ф.

Казань – 2019 г.

СОДЕРЖАНИЕ

ГЛОССАРИЙ

Анализ тональности текста - класс методов анализа текста, предназначенный для автоматизированного выявления в текстах эмоционально окрашенной лексики и эмоциональной оценки авторов по отношению к объектам, речь о которых идёт в тексте.

Датасет - выборка данных, предназначенных для анализа. Обычно в датасетах указываются различные классы и признаки каждого документа.

Искусственная нейронная сеть - математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей - сетей нервных клеток живого организма.

Корпус - коллекция текстов на естественном языке. Корпусы используются для проверки гипотез, обучения и статистического лингвистического анализа.

Метод обратного распространения ошибки - метод корректировки весовых параметров модели машинного обучения на основе величины ошибки модели на входных данных.

Стоп-слова - слова не несущие никакой смысловой или эмоциональной нагрузки, в их число входят различные местоимения и союзы.

Токен - элемент корпуса, им может быть слово, типографический символ или группа символов принимаемых за единый элемент текста.

Функция активации - функция, определяющая долю сигнала передаваемую нейроном или группой нейронов сети.

Эмотикон - пиктограмма, изображающая эмоцию. Чаще всего составляется из типографских знаков.

Эпоха обучения - одна итерация в процессе обучения, включающая предъявление всех примеров из обучающего множества и проверку качества обучения на контрольном множестве.

NLP (Natural Language Processing) - Обработка естественного языка - общее направление искусственного интеллекта и математической лингвистики. Оно изучает проблемы компьютерного анализа и синтеза естественных языков.

ВВЕДЕНИЕ

На сегодняшний день среди интернет ресурсов имеет большую популярность такая разновидность блогинга как микроблогинг, использующая концепцию коротких постов (100-200 слов). На популярных веб-сайтах, предоставляющих услуги микроблогинга (Twitter, Tumblr и Facebook), ежедневно появляются миллионы новых сообщений. Авторы этих сообщений пишут о своей жизни, обмениваются мнениями по различным темам и обсуждают актуальные проблемы. Поскольку пользователи часто высказываются о продуктах и услугах, которые они используют или же выражают свои политические и религиозные взгляды, веб-сайты микроблогов становятся ценными источниками мнений и настроений. Такие данные могут быть эффективно использованы для маркетинга или социальных исследований. Например, производственные компании могут быть заинтересованы в следующих вопросах:

- Какие ключевые аспекты продукта пользователи обсуждают?
- Насколько положительно или отрицательно люди относятся к нашему продукту?
- Какие функции продуктов пользователи хотят улучшить?

Политические партии могут быть заинтересованы в том, поддерживают ли люди их программу [1, 2]. Общественные организации могут узнать мнение людей о текущих дебатах.

Чтобы извлечь такую информацию необходимо анализировать внушительные объёмы текстовых данных. Решению этой проблемы посвящен специальный раздел компьютерной лингвистики – автоматический анализ тональности текста (sentiment analysis или opinion mining) [3].

Несмотря на множество решений данной проблемы, все они не идеальны и не обладают совершенной точностью в силу специфических особенностей текстовых сообщений в микроблогах: такие сообщения ограничены по длине, обычно охватывают одно предложение или меньше, а также содержат опечатки, сленг, эмотиконы и сокращенные формы слов.

Целью данной работы является разработка эффективного классификатора тональности коротких сообщений, опубликованных в сети Интернет, на основе современных методов глубокого обучения с учителем.

В связи с поставленной целью были поставлены следующие задачи:

- Провести обзор современных архитектур нейронных сетей (рекуррентные и сверточные нейронные сети) и эффективных методов обработки естественного языка (модели векторного представления слов, методы подсчета признаков на основе словарей тональности), применяемых для решения задачи классификации тональности.
- Выбрать программные средства для разработки и тестирования моделей классификации, спроектировать архитектуру системы.
- Разработать модели глубокого обучения на основе рассмотренных методов.
- Оценить качество моделей с различной конфигурацией на коллекции коротких сообщений социальной сети Твиттер и выбрать среди них наилучшую.

Объектом исследования является анализ тональности текстов на естественном языке.

Предметом исследования является классификатор на основе методов автоматического анализа тональности текста.

1. ОБЗОР МЕТОДОВ АНАЛИЗА ТОНАЛЬНОСТИ

Среди средств решения задачи классификации тональности текстов наиболее эффективными в течении последних лет являются методы машинного обучения, комбинированные с использованием векторного представления слов, а также методы основанные на использовании словарей тональной лексики [4, 5, 6].

В методах машинного обучения задача анализа тональности сводится к задаче классификации текстов, которая может быть решена путем обучения классификатора на заранее размеченной коллекции текстовых документов, относящихся к целевой или схожей предметной области [7].

Принцип подходов с использованием тональных словарей основан на определении тональности отдельных слов и последующую классификацию текста согласно полученным значениям [5]. Для этого используются специально построенные словари тональности, в которых для каждого слова обозначена соответствующая величина отражающая вес слова в контексте определенной тональности, к примеру позитивной или негативной.

Каждый из этих подходов имеет ряд преимуществ и недостатков. К примеру, для применения методов, основанных на использовании словарей тональности не требуется обучающая выборка и построение функции обучения. Таким образом, отнесение текста к определенному классу тональности может быть вполне легко объяснено [5]. С другой стороны, данные методы требуют построения заранее размеченных словарей тональности, в которых обязательно должна учитываться предметная область исследуемой коллекции текстов. При использовании методов машинного обучения, наличие словарей тональности не требуется. Большую значимость для данных методов имеет подходящая

обучающая коллекция и ее предобработка [8], в ходе которой может быть произведено приведение слов к общему виду, а также дополнение исходных данных вспомогательной информацией (обозначение фрагментов написанных в верхнем регистре или имеющих повторяющиеся символы, определение является ли последовательность хештегом, датой или обращением к другому пользователю и т.д.). Для методов машинного обучения также важен способ представления входных данных, обозначение признаков классификации, функция обучения и конфигурация глобальных параметров обучения классифицирующей модели. Одним из недостатков этих методов является тот факт, что классификатор, обученный на текстах одной предметной области, может быть не пригоден для использования в другой предметной области в силу низкого качества [7]. Стоит отметить, что на практике возникает сложность объяснения факторов повлиявших на отнесение текста к тому или иному классу.

Далее будет представлен обзор современных методов, используемых для решения задачи классификации тональности текстов.

1.1 Подходы к анализу тональности на основе моделей глубокого обучения

Глубокие нейронные сети за последнее десятилетие успешно применялись во многих прикладных задачах, которые ранее решались классическими алгоритмами машинного обучения [4, 9, 10]. В частности, в задачах обработки естественного языка, методами, показывающими наиболее высокое качество оказались рекуррентные, сверточные нейронные сети, а также их композиции [11, 12].

1.1.1 Рекуррентные нейронные сети

Большинство видов нейронных сетей предусматривают, что входы сети независимы друг от друга, однако зачастую в задачах связанных с текстовыми

или видео данными требуется обработка каждого отдельного входа в контексте предыдущего или следующего. Для таких задач существует особый вид нейронных сетей - рекуррентные нейронные сети (англ. recurrent neural network, RNN).

Главной особенностью рекурентных нейронных сетей является то, что они предназначены для обработки последовательных данных. Такие сети обрабатывают каждый элемент по отдельности, но при этом используют некоторую информацию оставшуюся от обработки предыдущего элемента последовательности [13]. Данный подход похож на то, как человек воспринимают текстовую информацию: при чтении люди понимают смысл каждого нового слова в предложении на основе ранее прочитанных слов и в конце предложения делают вывод по прочитанному. В общем виде графическое представление рекуррентной нейронной сети изображено на рисунке 1.

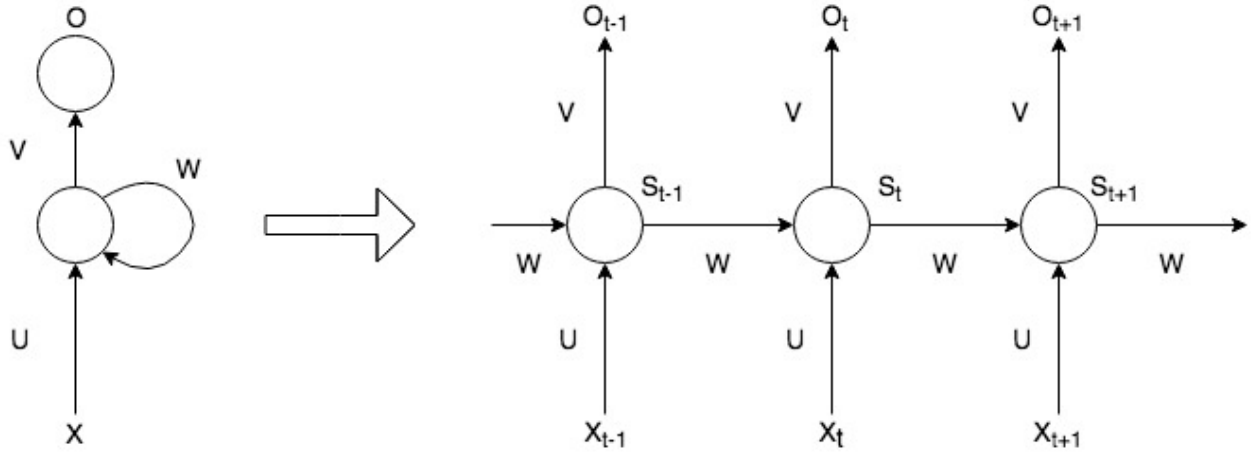

Рисунок 1. Схема рекуррентной нейронной сети

На приведенном выше рисунке показано, как RNN развертывается в полную сеть, то есть описывается для полной последовательности. Например, если последовательность, которую необходимо обработать это предложение из 5 слов, сеть будет развернута в 5-слойную нейронную сеть, по одному слою для каждого слова. Основные обозначения, используемые в вычислениях RNN можно описать следующим образом. Значение x_t является входным значением

на временном шаге t. Например, x_0 может быть вектором первого слова в предложении. s_t является скрытым состоянием или памятью сети на шаге t. В общем виде s_t вычисляется по следующей формуле:

$$s_t = f(Ux_t + Ws_{t-1} + b), \quad (1)$$

где:

- s_{t-1} - предыдущее значение скрытого состояния;
- U, W - матрицы весов скрытого слоя и скрытого состояния;
- f - функция активации, например ReLU или tanh;
- b - параметр отклонения.

Переменная o_t соответствует выходному вектору на шаге t. К примеру, в случае задачи предсказания следующего слова в последовательности им может быть вектор вероятностей слов из словаря подсчитанный с помощью функции $softmax$, то есть

$$o_t = softmax(Vs_t + c), \quad (2)$$

где:

- V - матрица весов выходного слоя;
- c - параметр отклонения.

Стоит заметить, что наличие выходного вектора на каждом шаге не обязательно, например в случае анализа тональности имеет значение лишь последнее выходное значение по которому будет сделано предсказание.

Рекуррентные модели с большим количеством слоев могут оказаться неустойчивыми: в силу многочисленных произведений на матрицу весов W при их корректировке во время обучения, значения обратно распространяемого градиента, как правило, могут исчезать или зашкаливать. Эти проблемы известны как проблемы затухающего (англ. vanishing gradient) и взрывающегося градиента (англ. exploding gradient), именно по их причине традиционные рекуррентные сети не способны запоминать долгосрочные зависимости в

последовательных данных [14]. Однако существует несколько вариантов RNN ячеек позволяющих этого избежать, одним из которых является Долгая краткосрочная память.

Долгая краткосрочная память (англ. Long Short Term Memory) – это разновидность архитектуры рекурентной нейронной сети, позволяющая гибко оперировать долгосрочными и краткосрочными зависимостями в последовательности за счет адаптивного обновления состояния ячейки (англ. cell state) [15].

Как видно из рисунка 2, принцип работы сети LSTM в точности удовлетворяет принципу работы RNN. Однако, существуют характерные особенности. Во-первых, в LSTM-сети отдельно выделяют ее состояние ячейки (англ. cell state) от которого зависит выход скрытого состояния h_t . Во-вторых, процесс обновления состояния на каждом шаге разбит на 2 этапа: забывание и обновление. За счет подобной конструкции LSTM-сеть может добавлять и удалять информацию в состояние ячейки.

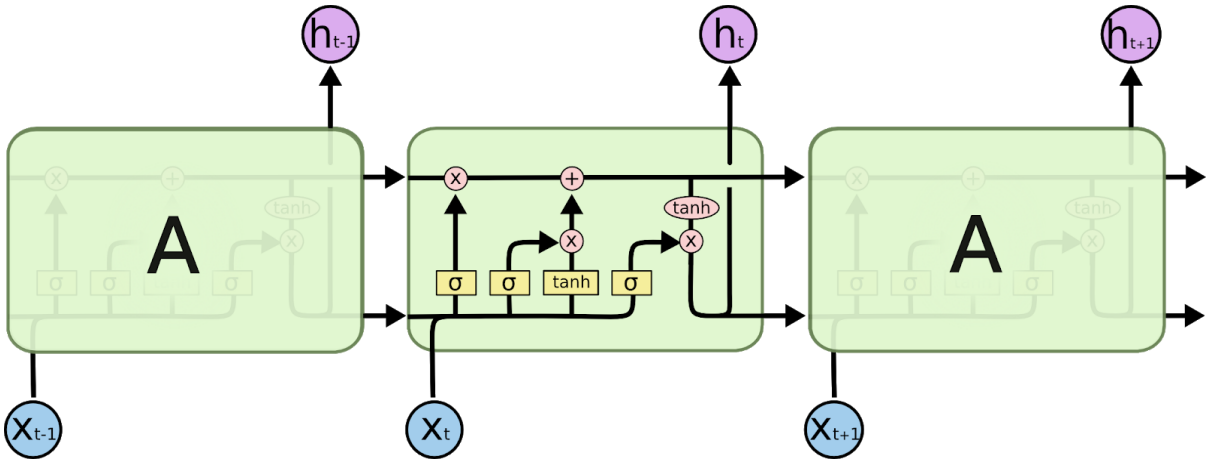

Рисунок 2. Архитектура ячейки LSTM

Источник: Olah C. Understanding LSTM Networks [16]

Рассмотрим принцип работы LSTM подробнее. На первом шаге ячейкой определяется какую информацию необходимо удалить из состояния ячейки. Это решение принимается слоем с сигмоидной функцией активации, называемым

“шлюзом забывания” (англ. forget gate). Он принимает значения h_{t-1} и x_t и выводит числа между 0 и 1 для каждого соответствующего числа состояния ячейки C_{t-1}:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f). \quad (3)$$

В случае задачи построения языковой модели, цель которой заключается в определении вероятности появления следующего слова по заданной последовательности слов, состояние ячейки может включать пол текущего субъекта, для возможности использовать правильные местоимения. В случае появления нового субъекта информация о предыдущем должна быть удалена.

Следующий шаг заключается в определении какая часть скрытого состояния будет сохранена в состоянии ячейки. Этот процесс состоит из двух этапов. Во-первых, сигмоидный слой, называемый “входным шлюзом” (англ. input gate), вычисляет, какие значения необходимо обновить. Затем слой с функцией активации *tanh* создает вектор новых значений-кандидатов C_t, которые могут быть добавлены в состояние ячейки:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \quad (4)$$

$$\widehat{C}_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c). \quad (5)$$

В языковой модели таким образом мог бы быть добавлена информация о поле нового субъекта к состоянию клетки, чтобы заменить предыдущий.

После этого происходит обновление старого состояния ячейки C_{t-1}, в новое состояние ячейки C_t на основе предыдущих шагов:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widehat{C}_t. \quad (6)$$

В случае языковой модели именно здесь отбрасывается информацию о поле старого субъекта и добавляется новая информация вычисленная на основе текущего входного значения и скрытого состояния.

Наконец, вычисляется вектор нового скрытого состояния. Его значение будет основано на текущем состоянии ячейки, но будет являться его отфильтрованной версией. Во-первых, еще один слой с сигмоидной функцией, называемый “выходным шлюзом” (англ. output gate), вычисляет, какие части состояния ячейки будут сохранены в скрытом состоянии. Затем значения вектора состояния ячейки проходят через функцию гиперболического тангенса и умножаются на выход “выходного шлюза”:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o), \quad (7)$$

$$h_t = o_t \odot tanh(c_{t-1}). \quad (8)$$

Для языковой модели, после обработки текущего значения субъекта, ячейка LSTM может вывести таким образом необходимую информацию для правильного построения формы следующего слова. Например, она может вывести, находится ли субъект в форме единственного или множественного числа, так что будет ясно каким образом надо спрягать следующий глагол.

1.1.2 Сверточные нейронные сети

Сверточные нейронные сети (англ. convolutional neural network, CNN) широко применяются в областях компьютерного зрения, но также позволяют достичь высоких показателей в задачах текстовой классификации [12, 17]. Для понимания принципа их работы на тексте сперва имеет смысл получить представление о том, как сверточные сети производят обработку графических данных, а также понять как происходят операции свертки при помощи фильтров.

Предположим, что имеется некоторое одноканальное изображение представленное в виде некоторой матрицы значений интенсивности канала. Типичный процесс свертки изображен на рисунке 3, его цель заключается в построении матрицы из суммы поэлементных произведений некоторой

матрицы весов называемой фильтром с группами пикселей изображения соразмерных этому фильтру. В процессе свертки происходят сдвиги фильтра по матрице изображения и в результате вычислений на каждом сдвиге образуется один из сигналов следующего слоя.

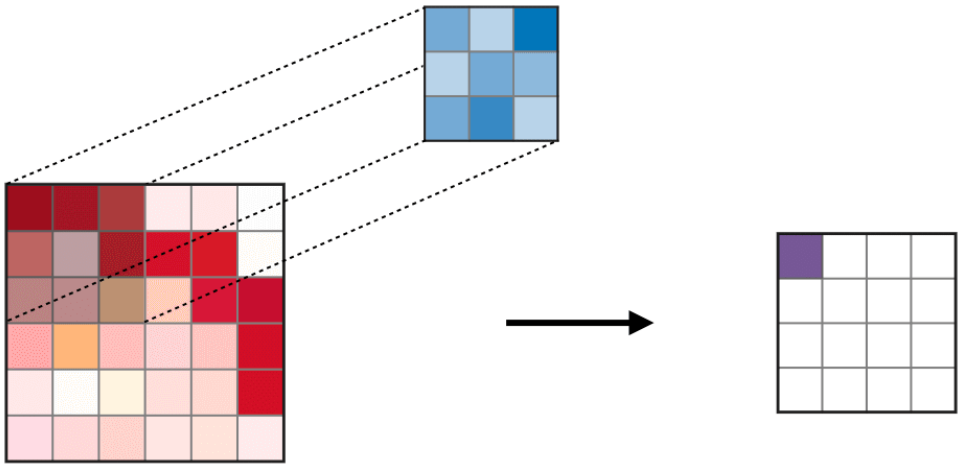

Рисунок 3. Схема процесса свертки в CNN.

Источник: A. Amidi, S. Amidi Convolutional Neural Networks cheatsheet [18]

CNN - это сеть представляющая собой несколько слоев сверток с нелинейными функциями активации, такими как ReLU или tanh, примененными к результатам [19]. В традиционной нейронной сети прямого распространения каждый входной нейрон соединен с каждым выходным нейроном в следующем слое, то есть каждый слой является полностью связным. В CNN соединение слоев происходит иначе. Вместо этого для вычисления выхода используются свертки над входным слоем. Это приводит к локальным соединениям, где каждая область входа подключается к нейрону на выходе. Каждый слой применяет различные фильтры, обычно сотни или тысячи, и объединяет их результаты. На этапе обучения CNN автоматически вычисляет значения своих фильтров на основе задачи, которую необходимо выполнить. Например, для задачи классификации изображений CNN может научиться обнаруживать ребра из необработанных пикселей в первом слое, затем использовать ребра для

обнаружения простых фигур во втором слое, а затем использовать эти фигуры для определения объектов более высокого уровня, таких как лица, в более высоких слоях. Последний слой может быть классификатором, использующим эти высокоуровневые объекты для предсказания класса изображения.

В случае текстовых задач, подобно матрице значений интенсивности каналов изображения, сетью используется матрица векторов слов, полученная при помощи алгоритмов векторного представления, например Word2Vec, Glove или же векторы полученные при помощи унитарного кодирования. Данная матрица будет иметь размерность $k \times n$, где k длина входной последовательности слов, а n размерность векторного пространства слов.

Фильтры для данной сети обычно имеют количество столбцов равное размерности векторов слов, а количество строк подбирается эмпирическим путем. Таким образом, при операции свертки фильтры движутся сверху вниз по группе векторов слов, охватывая несколько векторов подряд, и на них происходит вычисление выходных значений. Этот процесс проиллюстрирован на рисунке 4.

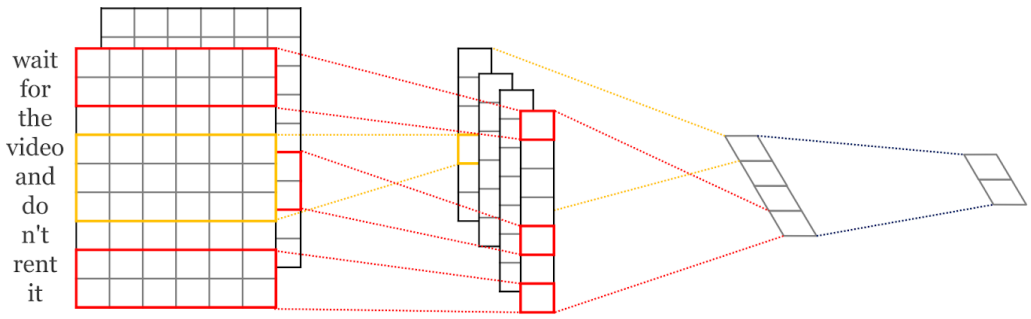

Рисунок 4. Процесс свертки в CNN для текстовых данных.

Источник: Kim Y. Convolutional neural networks for sentence classification [20]

Ключевым аспектом сверточных нейронных сетей является объединение слоев (англ. pooling), обычно применяемое после сверточных слоев. Объединяющие слои выделяют подвыборку из поступающих входных данных.

Наиболее распространенный способ объединения это взятие максимума из результата каждого фильтра.

Одним из свойств объединения является то, что оно предоставляет матрицу вывода фиксированного размера, которая обычно требуется для классификации. Например, если имеется 1000 фильтров и объединение по максимуму применяется к результатам каждой свертки, то получится 1000-мерный выход, независимо от размера фильтров или размера входа сети. Это позволяет использовать предложения и фильтры переменного размера, но всегда получать одинаковые выходные размеры для подачи в классификатор.

Объединение также уменьшает размерность вывода, сохраняя при этом самую важную информацию. Можно считать, что применение каждого фильтра определяет признаки данной группы слов. Одним из таких признаков может являться наличие отрицания, содержащееся в словосочетании “не подходящий”. Если эта фраза встречается в определенной области предложения, результат применения фильтра к этой области даст большое значение, но небольшое значение в других областях. Однако, стоит учесть, что несмотря на возможность определения обладает ли сообщение некоторым признаком при нахождении максимума, также теряется информацию о том, где именно был найден этот признак [19].

1.2 Векторные представления слов

В большинстве случаев в автоматизированных средствах анализа естественного языка минимальными анализируемыми единицами данных являются слова, и качество модели глубокого обучения зависит от способа их представления [21, 22]. Одним из простейших способов представления слов является унитарное кодирование (англ. one-hot encoding), суть которого заключается в сопоставлении слову i некоторого вектора длины k, где k число

всех слов в словаре анализируемого корпуса, так, что i-тая координата вектора равна единице, а все остальные нулям. При простоте реализации такой подход обладает двумя важными недостатками: при большом объеме словаря такие векторы будут неоптимальными для хранения в памяти, так как большая часть пространства будет заполнена нулями, а также будет невозможно сказать насколько те или иные слова различаются по смыслу.

Однако, существуют различные методы отображения множества слов на пространство вещественных чисел $\mathbb{R}^n$ таким образом, что геометрические соотношения между векторами будут представлять семантическую близость соответствующих слов, при возможности определения размерности векторов значительно меньше объема словаря. Данные методы известны под общим названием как векторное представление слов (англ. Word Embeddings). Их идея основывается на предположении, что семантически близкие слова часто встречаются в схожих контекстах. Исходя из этого рассуждения, для контекста отзывов на продукты питания слова “хлеб” и “молоко” будут семантически близки, а векторы этих слов будут иметь относительно большое значение косинусной меры. Далее будет представлен обзор на несколько эффективных алгоритмов векторного представления слов.

1.2.1 Word2Vec

Word2Vec представляет собой набор алгоритмов векторного представления слов и был представлен группой исследователей Google в 2013 году [23]. В группу алгоритмов Word2Vec входят два типа архитектур нейронной сети: Continuous Bag-of-Words (CBOW) и skip-gram. Первая архитектура предназначена для определения целевого слова на основе слов, используемых с ним в контексте: например это может быть слово “подоконник” в контексте слов “кот”, “сидеть”, “на”. Статистически это приводит к тому, что

CBOW сглаживает большую часть распределительной информации, рассматривая каждый контекст как одно наблюдение. В основном это оказывается полезным для небольших наборов данных. Архитектура skip-gram, в свою очередь, предназначена для обратной задачи - определения контекста слов на основе определенного входного слова. Для решения этой задачи, skip-gram рассматривает каждую пару “контекст-слово” как новое наблюдение, и это, как правило, работает лучше при анализе больших объемов данных. Алгоритмически, обе модели близки, их представление схематически изображено на рисунке 5.

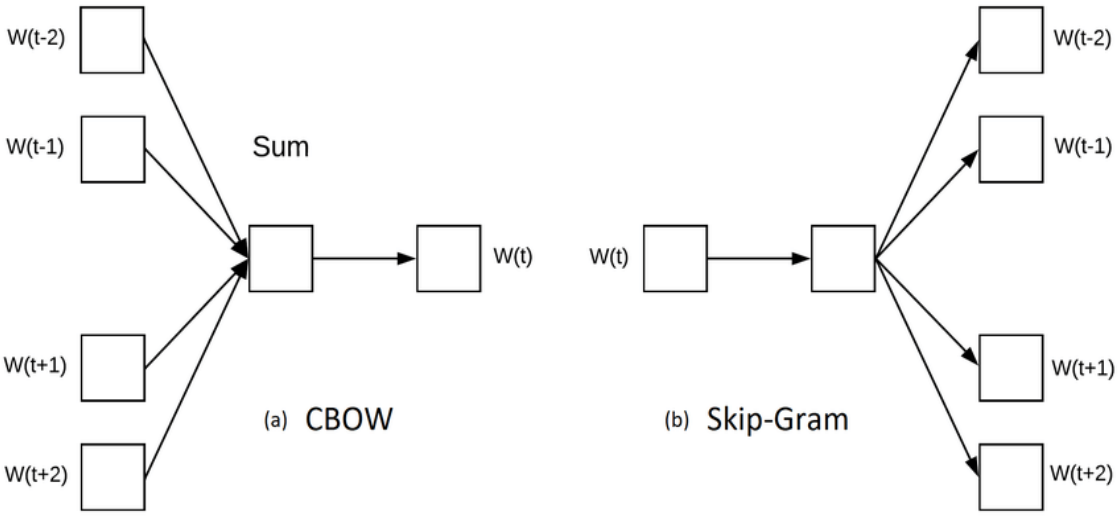

Рисунок 5. Высокоуровневое представление моделей CBOW и skip-gram

Источник: Çano E. Text-based sentiment analysis and music emotion recognition [24]

Рассмотрим принцип работы алгоритмов Word2Vec на примере модели skip-gram:

- Определяется некоторый корпус D с величиной словаря V .
- Определяется размерность векторного пространства слов N. Теоретически данное число может быть произвольным положительным числом меньшим V , но на практике большая размерность векторов способствует лучшему качеству модели [23].

- Определяется размер окна контекста C соответствующий количеству слов находящихся в области рассматриваемого слова включая само слово, находящимся посередине. То есть, для фразы “сегодня была отличная погода”, контекст размера 3 для слова “отличная” будет включать слова “была” и “погода”.
- Произвольным образом определяются матрицы весов входного и выходного слоя W_1 и W_2, размерностей $V \times N$ и $N \times V$ соответственно.
- Определяется количество итераций обучения модели - циклов предсказания и корректировки весов на основе полученной ошибки.
- На вход модели подается one-hot вектор некоторого целевого слова w_t и группа векторов слов контекста целевого слова $w_{c1}, ..., w_{ck}$.
- Вычисляется вектор скрытого слоя h, равный $w_t * W_1$ и затем же вычисляется вектор выходного слоя o, равный $h * W_2$.
- На основе выходного вектора вычисляется апостериорная вероятность принадлежности слов к контексту целевого слова с помощью функции $softmax$.
- Для каждого слова текущего контекста высчитывается значение ошибки и происходит обновление весов матриц с помощью метода обратного распространения.
- Контекстное окно слов сдвигается на слово вперед, цикл обучения повторяется.
- После указанного количества итераций обучение прерывается. Финальные строки матрицы W_1 будут являться векторами слов, где i-я строка матрицы соответствует i-му слову словаря проанализированного корпуса.

1.2.2 GloVe

Еще одной эффективной моделью векторного представления слов является модель GloVe (Global Vectors), предложенную исследователями стэнфордского университета в 2014 году [25]. В отличии от основанной на предсказании модели Word2Vec, GloVe основывается на подсчете совместного появления слов всего корпуса. Формально, цель модели GloVe заключается в построении векторов подчиняющихся следующему ограничению:

$$\overline{w}_i^T \overline{w}_j + b_i + b_j = logX_{ij}, \tag{9}$$

где:

- $\overline{w}_i$, $\overline{w}_j$ - вектора слов i, j соответственно;
- b_i, b_j - величины смещения для слов i и j;
- X_{ij} - количество появлений слова j в контексте слова j.

Интуитивно, целью модели GloVe является построение векторов слов, которые сохраняют некоторую полезную информацию о каждом совместном появлении слов i и j. Данная цель достигается в GloVe с помощью минимизации целевой функции ошибки:

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(\overline{w}_i^T \overline{w}_j + b_i + b_j - logX_{ij})^2, \tag{10}$$

где, f - специальная функция взвешивания GloVe:

$$f = \begin{cases} \left(\frac{X_{ij}}{x_{max}}\right)^{\alpha}, & если\ X_{ij} < x_{max} \\ 1, & иначе \end{cases}. \tag{11}$$

Без наличия данной функции, модель будет одинаково взвешивать все совпадения, даже те, которые случаются редко. Такие редкие совпадения являются шумными и несут меньше информации, чем более частые. Функция взвешивания обладает двумя параметрами:

1. x_{max} - параметр определяющий порог количества совместной встречаемости слов X_{ij}. При превышении этого порога величина ошибки для данного количества встречаемостей остается неизменной.
2. α - параметр с помощью которого определяется то, насколько будет уменьшен вес ошибки в случае если количество встречаемостей ниже порога x_{max}.

В работе, описывающей подход GloVe, качество модели было оценено на задаче аналогии слов, цель которой заключается в выборе слова наилучшим образом отвечающим на вопрос вида "a для b как c для _?". Тестовый набор данных использованный для оценки содержал 19544 таких вопроса, разделенных на семантическое и синтаксическое подмножества. Семантические вопросы являются вопросами вида ""Афины" для "Греция", как "Берлин" для _?", а синтаксические представляют собой вопросы вида ""танцевать" для "танец", как "петь" для _?". По результатам оценки нескольких моделей GloVe в среднем показала лучшие результаты на обоих множествах, существенно превосходя представления Word2Vec и другие аналоги [25].

1.2.3 ELMo

Очевидно, что большую значимость для смысла слова имеет то, в каком контексте оно использовано. Вышеописанные модели векторного представления Word2Vec и GloVe позволяют эффективным образом построить вектора слов в семантическом пространстве, однако при рассмотрении некоторого предложения или группы связанных предложений в виде последовательности этих векторов, отдельно взятый вектор не будет содержать в себе информацию о контекстуальном смысле соответствующего слова. В 2018ом году в области NLP были предложены различные модели векторного

представления слов, в которых отсутствует данный недостаток [26, 27]. Одной из таких моделей является модель ELMo (Embeddings from Language Models) разработанная исследователями проекта AllenNLP.

Вместо того, чтобы использовать фиксированный вектор для каждого слова, модель ELMo анализирует все предложение или абзац целиком, вычисляя соответствующие вектора слов для каждой отдельной последовательности. При этом в каждом векторе сохраняется информация о контексте, в котором было использовано слово. Для этого используется модель, использующая двунаправленные слои LSTM и предобученная на задаче моделирования языка.

Также стоит отметить, что представления ELMo основаны на символах. Это позволяет сети использовать морфологические признаки для формирования качественных представлений для словарных лексем, которые отсутствовали в обучении.

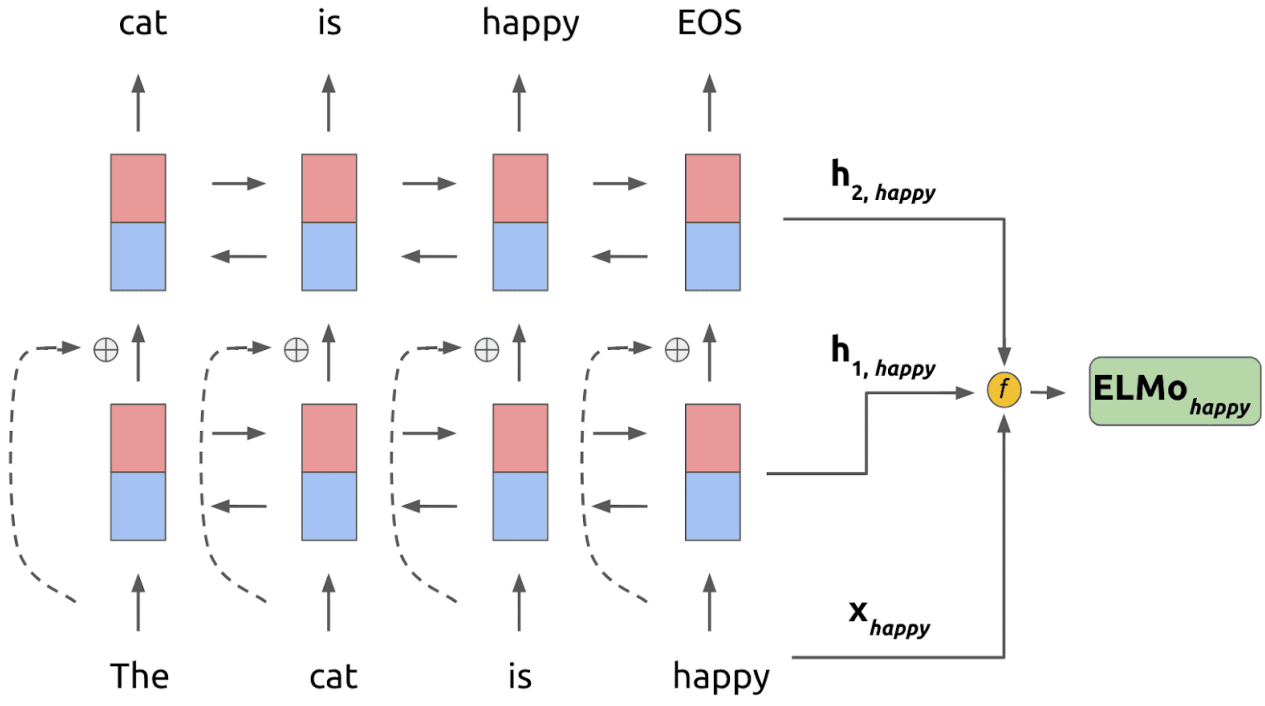

Рисунок 6. Представление модели ELMo

Источник: M. Eric Deep Contextualized Word Representations with ELMo [28]

Формально, в модели ELMo, представления слов вычисляются с помощью следующего уравнения:

$$ELMo_k = \gamma \sum_{j=0}^{L} s_j h_{k,j} . \quad (12)$$

Здесь $h_{k,j}$, $j = \overline{1, L}$ являются выходными значениями LSTM слоя под индексом j для kго слова, а $h_{k,0}$ значениями входного слоя. Параметры s_j и γ являются весами определяемыми во время обучения и зависят от определенной задачи, причем s_j позволяет масштабировать определенные значения $h_{k,j}$, в то время как γ позволяет масштабировать весь выходной вектор целиком.

По результатам испытаний, использование модели ELMo значительно улучшило качества современных моделей применяемых для создания вопросно-ответных систем, семантической ролевой маркировки, анализа тональности и других задач NLP [26].

1.3 Признаки тональности, используемые для классификации текстов

В случае ограниченного количества имеющихся данных, помимо признаков, выявленных обучаемой моделью, для классификации тональности сообщения также могут использованы вспомогательные признаки подсчитанные на анализируемом корпусе до обучения [5]. Данные признаки могут быть подсчитаны на основе словарей тональности, значения которых могут быть собраны вручную или подсчитаны автоматически на размеченном корпусе данных [29].

В общем случае тональные словари представляют собой списки слов (и фраз) с соответствующими им классами тональности. Некоторые словари могут дополнительно содержать оценку настроения для обозначения силы его интенсивности. Более высокие значения настроения указывают на большую

интенсивность. Например, запись “great” (positive, 1.2) означает, что слово “great” имеет положительную тональность с оценкой интенсивности 1.2. Запись “acceptable” (positive, 0.1) указывает, что слово “acceptable” также имеет положительную тональности и его интенсивность ниже, чем у слова “great”.

К достоинствам словарей тональности построенных вручную можно отнести их качество полученное с помощью человеческой разметки, однако построение таких словарей является трудоемкой задачей по причине самостоятельной оценки данных человеком, которая может требовать большого количества времени. Помимо этого значения таких словарей не обладают точной оценкой интенсивности тональности и содержат лишь конкретные обозначения класса (например “negative” или “strong negative”).

Для автоматического построения словарей тональности используются методы оценивающие степень взаимосвязи слова с конкретным классом. В исследовании [5] для определения оценки тональности слова была использована мера поточечной взаимной информации (англ. Pointwise mutual information), значение которой для позитивного класса тональности на имеющийся коллекции текстов может быть подсчитано следующим образом:

$$PMI(w, positive) = \log_2 \frac{freq(w, positive) \cdot N}{freq(w) \cdot freq(positive)}. \quad (13)$$

Здесь $freq(w, positive)$ это количество появлений слова w в сообщениях позитивного класса, N общее количество токенов корпуса, $freq(w)$ количество появлений слова w в корпусе и $freq(w, positive)$ общее количество слов в сообщениях относящихся к позитивному классу. С помощью этого значения оценка тональности определенного может быть подсчитана на основе следующего выражения:

$$Sentiment\ Score(w) = PMI(w, positive) - PMI(w, negative). \quad (14)$$

Стоит отметить что помимо взаимной точечной информации, также может быть использована функция перекрестной энтропии или значение прироста информации. Также следует учитывать, что для эффективности данных методов необходимо наличие объемной размеченной выборки.

1.4 Метрики качества классифицирующей модели

Для оценки качества классифицирующей модели в первую очередь необходимо наличие тестовой выборки с заранее размеченными верными классами документов. Для того чтобы определить насколько хорошо или плохо модель определяет классы на тестовой выборке могут быть использованы различные метрики качества. Наиболее простой из них является аккуратность (англ. accuracy). Аккуратность - это отношение количества элементов выборки, в которых классификатор верно определил класс, к общему количеству ее элементов. Несмотря на простоту, данная метрика может быть бесполезна в случае, когда число документов обучающей выборки для одного класса сильно смещено относительно других классов: это может привести к тому, что классификатор будет неправильно предсказывать классы для которых имелось мало информации во время обучения, но аккуратность будет иметь высокое значение, так как класс для которого имелось больше документов предсказывается моделью гораздо лучше.

На практике чаще используются следующие метрики качества по каждому из классов: точность (англ. precision), полнота (англ. recall) и F-мера (англ. F-score). Для определения каждого из них стоит ввести следующие понятия:

- Истинно-положительные решения (англ. True Positive, TP) - число документов, для которых текущий предсказанный класс соответствует действительному.

- Ложно-положительные решения (англ. False Positive, FP) - число документов, для которых текущий предсказанный моделью класс оказался ошибочным.
- Ложно-отрицательные решения (англ. False Negative, FN) - число документов, которые должны были быть отнесены к текущему классу, но ошибочно были отнесены к другим.

Имея следующие обозначения, точность и полнота могут быть определены следующим образом:

$$P = \frac{TP}{TP + FP}, \tag{15}$$

$$R = \frac{TP}{TP + FN}. \tag{16}$$

По своей сути, точность описывает число верно предсказанных документов относительно всех документов причисленных моделью к этому классу, а полнота - число правильно классифицированных документов относительно всех документов этого класса. Для объединения значений двух мер в одну может быть использована F-мера, являющаяся гармоническим средним этих величин:

$$F = 2\frac{P * R}{P + R}. \tag{17}$$

При анализе качества на нескольких датасетах могут быть использованы микро оценки качества модели - микро-точность, микро-полнота и микро-F мера. Формулы для вычисления микро-точности и микро-полноты:

$$P_{micro-average} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + FP_i}, \tag{18}$$

$$R_{micro-average} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + FN_i}. \tag{19}$$

Для анализа результатов работы модели для всех классов в среднем могут быть использованы макро метрики:

$$P_{macro-average} = \frac{\sum_{i=1}^{C} P_i}{C}, \quad (20)$$

$$R_{macro-average} = \frac{\sum_{i=1}^{C} R_i}{C}. \quad (21)$$

В случае когда множества документов каждого класса значительно отличаются по объему могут быть использованы взвешенные макро метрики для учета дисбаланса классов в выборке.

2. ПРОЦЕСС РАЗРАБОТКИ КЛАССИФИКАТОРОВ

2.1 Выбор данных для обучения и тестирования

Для обучения и тестирования классифицирующей модели был выбран датасет соревнований международного практикума по семантическим оценкам SemEval 2017 для задачи классификации тональности сообщений. Данный датасет состоит из множества сообщений опубликованных в социальной сети Twitter в период с 2013 года по 2016 год, разделенных на 3 класса тональности: негативный, нейтральный и позитивный.

Таблица 1. Количество документов датасета соревнования SemEval для задачи оценки тональности сообщений.

	Количество сообщений с негативным классом тональности	Количество сообщений с нейтральным классом тональности	Количество сообщений с позитивным классом тональности	**Общее количество документов**
Обучающая выборка	7840	22591	19902	**50333**
Тестовая выборка	3972	5937	2375	**12284**

Набор данных включает себя твиты собранные по различным темам, содержащим как именованные сущности (например, Дональд Трамп, iPhone), так и геополитические сущности (например, Аллеппо, Палестина).

Разметка данных происходила аннотаторами платформы CrowdFlower (сейчас Figure Eight). Каждый твит был оценен как минимум пятью аннотаторами по пятибалльной шкале: от крайне негативной оценки тональности до крайне позитивной. Если 3 аннотатора из 5 имели одинаковую

оценку, то в качестве результирующего класса принималась эта оценка, в противном случае значения оценок усреднялись и округлялись до ближайшего целого числа.

2.2 Инструменты для разработки

В качестве основного инструмента для разработки классификаторов был выбран высокоуровневый язык программирования общего назначения Python 3й версии, в силу его популярности и эффективности в задачах анализа данных и машинного обучения. Для ускорения процесса разработки были использованы следующие библиотеки Python:

- NumPy - библиотека, добавляющая поддержку многомерных массивов и матриц, вместе с множеством высокоуровневых математических функций для операций с этими массивами. В данном проекте библиотека использовалась для первичного хранения данных, таких как векторы слов и векторы тональных признаков.
- Pandas - библиотека для обработки и анализа данных построенная поверх библиотеки NumPy. Использовалась для импорта и хранения входных текстовых данных.
- Scikit-learn - библиотека языка Python, содержащая набор инструментов для задач машинного обучения. В данном проекте использовалась для подсчета метрик качества на тестовом датасете.
- Keras - высокоуровневая библиотека для задач глубокого обучения, предоставляющая средства для построения, обучения и анализа моделей глубокого обучения. Все модели в данном проекте строились с помощью Keras.
- TensorFlow - библиотека для задач машинного обучения, разработанная компанией Google для решения задач построения и обучения нейронных

сетей. Использовалась в качестве backend’а для библиотеки Keras, то есть с её помощью производились такие операции как свертка и произведение матрицы весов модели.

- Ekphrasis - библиотека предобработки текстовых данных, ориентированная на работу с текстами из социальных сетей.

2.3 Предобработка данных

Как упоминалось ранее, предобработка данных значительно влияет на качество классифицирующей модели [8]. Предобработка входных текстовых данных в разработанных моделях производилась следующим образом:

- Все буквы были приведены к нижнему регистру, знаки препинания отделялись знаками пробела от слов, для рассмотрения в качестве отдельных токенов.
- Орфография слов с опечатками корректировались на основе статистик корпусов Википедии и Твиттера используемых в библиотеке Ekphrasis.
- Сокращения вида “can’t” и “don’t” приводились к виду “can not” и “do not” соответственно.
- Обращения к пользователям, даты, указания времени, URL, номера телефонов, значения эмотиконов и слова с намеренно повторяющимися символами или состоящие только из заглавных букв помечались специальными тегами вида <user>, <date>, <time>, <url>, <happy>, <elongated>, <allcaps> и т.д.
- Хэштеги разбивались на отдельные слова и так же помечались специальными тегами.
- Слова с намерено повторяющимися символами и повторяющиеся знаки препинания приводились к общему виду.

Таблица 2. Примеры результатов предобработки сообщений.

Hmmm. so it may not be Ric Flair after all. someone else will be the new judge. IS IT STONE COLD STEVE AUSTIN? :O	hmm <elongated> . so it may not be ric flair after all . someone else will be the new judge . <allcaps> is it stone cold steve austin </allcaps> ? <surprise>
@AdamORogers @GovernorPerry @SCGOP Great questions & answers. Those there got to see the 10th Amendment Rick Perry we all love! #Perry2016	<user> <user> <user> great questions & answers . those there got to see the 1 0 th amendment rick perry we all love ! <hashtag> perry 2016 </hashtag>
"#Denali, the great one soaring under the midnight sun". Sarah Palin Sunday July 26, 2009 3pm Farewell Speech.	" <hashtag> denali </hashtag> , the great one soaring under the midnight sun " . sarah palin sunday <date> <time> farewell speech .

2.3 Разработка моделей

В качестве первоначальной базовой модели было решено использовать нейронную сеть из нескольких слоев LSTM и Embedding слоем, предоставляемым библиотекой Keras, и постепенно модифицировать эту сеть. В качестве первой модификации в модель были добавлены вектора слов, предобученные алгоритмом GloVe на множестве сообщений из Твиттера.

Также в модель были добавлены словарные признаки тональности, полученные на основе метода предложенного в работе [5]. Для их подсчета было использовано 5 словарей тональности:

- Hashtag Lexicon - словарь, автоматически построенный на основе твитов с эмоционально окрашенными хэштегами [5]. Содержит оценки тональности в утвердительном и отрицательном контексте отдельно для единичных токенов и последовательных пар токенов. Для подсчета значений тональности в нем используется мера поточечной взаимной информации.
- Sentiment 140 Lexicon - словарь, автоматический построенный на основе твитов с эмотиконами [30]. Также как и Hashtag Lexicon

содержит оценки в утвердительном и отрицательном контексте для единичных токенов и последовательных пар токенов. Для подсчета значений тональности в нем используются значения эмотиконов .

- MPQA Subjectivity Lexicon - словарь слов используемых для субъективной оценки, также содержит оценки тональности [31].
- NRC Emotion Lexicon - словарь тональности общего назначения построенный вручную с помощью сервиса Mechanical Turk и содержащий оценку эмоциональной окраски слов и их тональности [32].
- Bing Liu Lexicon - словарь тональности общего назначения, представляющий собой списки из позитивных и негативных слов [33].

При помощи метода, описанного в статье [34], сперва были обозначены токены, сигнализирующие о начале или конце отрицательного контекста. Например, такие слова как “never”, “no”, “cannot” или слова имеющие частицу “n’t” рассматривались как отрицания и все следующие после них токены принимались токенами негативного контекста. Некоторые знаки препинания, такие как точка, скобка, вопросительные и восклицательный знак, а также слово “but” рассматривались в качестве токенов, сигнализирующих о конце отрицательного контекста. Далее, во входных сообщениях выделялись токены находящиеся в непосредственном негативном контексте и удаленном негативном контексте: токенами непосредственного негативного контекста являются токены, непосредственно следующие за словом отрицания, а токенами удаленного контекста, являются все оставшиеся токены, идущие до токена обозначающего конец контекста отрицания. Токены непосредственного и удаленного отрицательного контекста помечались постфиксами “_NEGFIRST” и “_NEG” соответственно, знаки препинания и стоп-слова пропускались.

Таблица 3. Примеры сообщений с помеченными отрицательными контекстами.

<user> you can not beat_NEGFIRST a bit_NEG off michael_NEG jackson_NEG on a wednesday_NEG morning_NEG #time_NEG tunnel_NEG
<user> . come on . u mean to tell me u would not prefer_NEGFIRST to act_NEG alongside_NEG maite_NEG once again for the 3 rd_NEG time_NEG ? u just do not want_NEGFIRST ppl_NEG talking_NEG
the crown inn , lechlade summer beer festival starts tonight ! & do not miss_NEGFIRST lechlade_NEG cricket_NEG club_NEG big_NEG car_NEG boot_NEG sale_NEG on monday_NEG at <time>

После пометки контекстов происходил подсчет значений векторов признаков для каждого словаря тональности. Для словарей тональности построенных автоматически были подсчитаны следующие признаки:

- Количество токенов с ненулевой оценкой тональности.
- Сумма оценок тональности всех токенов.
- Максимальное значение оценки тональности среди всех токенов.
- Значение оценки тональности последнего токена.

Для словарей, построенных вручную, значения признаков вычислялись иначе, ниже представлен список этих значений:

- Сумма токенов с положительной оценкой тональности в утвердительном контексте.
- Сумма токенов с негативной оценкой тональности в утвердительном контексте.
- Сумма токенов с положительной оценкой тональности в отрицательном контексте.
- Сумма токенов с отрицательной оценкой тональности в отрицательном контексте.

Помимо модели с однонаправленными слоями LSTM, также использовалась модель с двунаправленными LSTM слоями и механизмом внимания (англ. Attention mechanism), построенная на примере данной работы [35].

Для последней модели была использована архитектура на основе модели векторного представления слов ELMo и нескольких сверточных слоев.

Стоит также отметить, что в каждую модель добавлялись отсеивающие слои (англ. Dropout) позволяющие предотвратить проблему переобучения. При использовании таких слоев случайно выбранные нейроны игнорируются во время обучения. Это означает, что их вклад в активацию последующих нейронов временно удаляется.

3. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Результаты обучения построенных моделей сравнивались с результатами моделей соревнования SemEval для задачи анализа тональности на сообщениях Твиттера [4]. В таблице 3 представлены метрики наилучших моделей SemEval, в которую входят ранее описанные макро-средние и взвешенные метрики.

Таблица 3. Результаты работы пяти лучших моделей соревнования SemEval

Название команды	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
DataStories	0.655	0.681	0.654	0.674	0.651	0.645
BB_twtr	0.674	0.681	0.660	0.685	0.658	0.651
LIA	0.662	0.676	0.660	0.674	0.661	0.657
Senti17	0.649	0.674	0.651	0.669	0.652	0.648
NNEMBs	0.654	0.669	0.660	0.667	0.664	0.664

Самой первой испытывалась многослойная модель без предобученных векторов слов и минимальной предобработкой текста (приведение токенов к нижнему регистру, удаление стоп-слов, пометка эмотиконов, обращений к пользователям, URL и т.п.).

Таблица 4. Результаты экспериментов на базовой LSTM модели

Модель	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
Emb. + LSTM (x3)	10	0.507	0.525	0.492	0.533	0.521	0.524

Emb. + Dropout (0.3) + LSTM (x3)	10	0.515	0.536	0.518	0.540	0.526	0.528

Для последующих экспериментов использовалась модифицированная версия данной модели с использованием векторных представлений GloVe и улучшенной предобработкой, описанной ранее. В ходе экспериментов менялось количество эпох обучения, доля отсеивания нейронов, наличие добавочного полносвязного слоя и другие параметры.

Таблица 5. Результаты экспериментов на LSTM модели с улучшенной предобработкой и векторными представлениями GloVe

Модель	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
Emb. (Glove) + Dropout (0.5) + LSTM x3	10	0.634	0.646	0.625	0.649	0.643	0.644
Emb. (Glove) + Dropout (0.3) + LSTM x3	15	0.613	0.608	0.609	0.621	0.619	0.618
Emb. (Glove) + Dropout (0.5) + LSTM x3	10	0.641	0.646	0.636	0.657	0.645	0.645
Emb. (Glove) + Dropout (0.2) + LSTM x3	10	0.626	0.637	0.631	0.639	0.637	0.637
Emb. (Glove) + Dropout (0.5) + LSTM x3 + Dense 100	10	0.645	0.659	0.651	0.656	0.653	0.653

Далее тестировалась та же модель, но с добавлением признаков словарей тональности на основе вышеописанного метода. Как видно из таблицы 5,

добавление дополнительных признаков незначительно улучшило некоторые показатели качества классификации.

Таблица 6. Результаты экспериментов на LSTM модели с использованием признаков тональных словарей

Модель	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
Emb. + Dropout (0.5) + LSTM x3 + conc. (MPQA, NRC, BingLiu) + Dense 100	10	0.649	0.649	0.645	0.658	0.654	0.653
Emb. + Dropout (0.5) + LSTM x3 + conc. (MPQA, NRC, BingLiu, Sent140, hashtag) + Dense 100	10	0.655	0.651	0.652	0.661	0.66	0.659

Для проверки устойчивости многослойной LSTM модели на данных другой области, были проведены эксперименты с использованием дополнительного корпуса комментариев на новостном веб-сайте Yahoo News - YNACC (The Yahoo News Annotated Comments Corpus) [35]. Из данного множества сообщений было извлечено 17,343 сообщения с рассматриваемыми классами, размеченных профессиональными аннотаторами. С полученным датасетом были произведены эксперименты, результаты которых представлены на таблице 8. Как видно из результатов экспериментов, в случае объединения корпуса YNACC с корпусом SemEval, ухудшение качества модели является незначительным, в отличии от случаев когда модель обучалась по очереди на двух датасетах или была обучена только на YNACC.

Таблица 7. Результаты экспериментов на LSTM с корпусом отзывов YNACC

Модель	Датасеты	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
Emb. (Glove) + Dropout (0.3) + LSTM x3 + Dense 100	SemEval, объединенный с YNACC	10	0.636	0.644	0.639	0.646	0.645	0.645
Emb. (Glove) + Dropout (0.3) + LSTM x3 + Dense 100	1.SemEval 2.YNACC	10	0.427	0.368	0.341	0.441	0.406	0.382
Emb. (Glove) + Dropout (0.3) + LSTM x3 + Dense 100	1.YNACC 2.SemEval	10	0.565	0.520	0.496	0.578	0.550	0.517
Emb. (Glove) + Dropout (0.3) + LSTM x3 + Dense 100	YNACC	10	0.555	0.459	0.418	0.551	0.455	0.418

Таблица 8. Результаты экспериментов на двунаправленной LSTM модели с механизмом внимания

Модель	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
Emb. + Dropout (0.5) + Bidirectional LSTM x2 + Attention	10	0.633	0.662	0.642	0.653	0.644	0.645
Emb. + GaussianNoise (0.3) + Dropout (0.3) + Bidirectional LSTM x2 + Dropout (0.3) +	40	0.596	0.616	0.6	0.618	0.606	0.608

Attention + Dropout (0.5)							

Однако, наиболее высоких показателей качества удалось добиться на модели, комбинированной из нескольких сверточных слоев и модели векторного представления ELMo. Результат второго эксперимента таблицы 6 демонстрирует значительное превосходство (+0.02 F1) по сравнению с каждой из моделей таблицы 3.

Таблица 9. Результаты экспериментов на модели с использованием сверточных слоев и векторных представлений ELMo

Модель	Кол-во эпох	macro-average P	macro-average R	macro-average F1	weighted P	weighted R	weighted F1
ELMo + (Conv1d, Maxpooling1d, Flatten) x3 + Dropout (0.6) + Dense 50	10	0.541	0.535	0.519	0.561	0.546	0.536
ELMo + (Conv1d, Maxpooling1d, Flatten) x3 + Dropout (0.4) + class weighting	35	0.677	0.690	0.681	0.691	0.687	0.688

ЗАКЛЮЧЕНИЕ

В ходе данной работы были выполнены следующие задачи:

- Были рассмотрены архитектуры рекуррентных и сверточных нейронных сетей, в том числе была рассмотрена разновидность ячейки рекуррентных нейронных сетей LSTM.
- Были рассмотрены методы векторного представления слов Word2Vec, GloVE и ELMo.
- Был рассмотрен метод классификации тональности на основе словарей тональности, построенных вручную и автоматически.
- На основе рассмотренных методов и выбранных инструментов разработки были построены модели классификации тональности различной архитектуры.
- Разработанные модели были апробированы на данных соревнования по семантическому анализу SemEval 2017 для задачи анализа тональности сообщений социальной сети Твиттер и сравнены с лучшими моделями данного соревнования.

По результатам экспериментов с имеющимися моделями была выявлена модель, обладающая наилучшими показателями - модель на основе нескольких сверточных слоев и модели векторного слов ELMo, для которой были получены оценки качества, превосходящие оценки моделей соревнования SemEval.

Таким образом цель данной работы была достигнута. Программный код работы и результаты экспериментов можно найти в открытом доступе по ссылке: http://gititis.kpfu.ru/marsel.mustafin/sentiment-nn

СПИСОК ЛИТЕРАТУРЫ

1. Mohammad S. et al. Semeval-2016 task 6: Detecting stance in tweets // Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). – 2016. – P. 31-41.
2. Krejzl P., Steinberger J. UWB at SemEval-2016 task 6: stance detection // Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). – 2016. – P. 408-412.
3. Liu B. et al. Sentiment Analysis and Subjectivity // Handbook of natural language processing. – 2010. – Т. 2. – №. 2010. – P. 627-666.
4. SemEval-2017 task 4: Sentiment analysis in Twitter. // Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017). – 2017, P. 502–518
5. Kiritchenko S., Zhu X., Mohammad S. M. Sentiment analysis of short informal texts // Journal of Artificial Intelligence Research. – 2014. – Т. 50. – P. 723-762.
6. Плетнева М.В. Программная система анализа тональности текстов на основе словарей оценочной лексики // Программная инженерия. – 2019. – Т. 10. – № 1. – С. 38-46.
7. Liu B. Sentiment analysis and opinion mining // Synthesis lectures on human language technologies. – 2012. – Т. 5. – №. 1. – P. 1-167.
8. Symeonidis S., Effrosynidis D., Arampatzis A. A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis // Expert Systems with Applications. – 2018. – Т. 110. – P. 298-310.
9. Zhang L., Wang S., Liu B. Deep learning for sentiment analysis: A survey //Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. – 2018.
10. Neethu M. S., Rajasree R. Sentiment analysis in twitter using machine learning techniques // 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT). – IEEE, 2013. – P. 1-5.
11. Explaining recurrent neural network predictions in sentiment analysis. // Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. – 2017 – P. 159–168

12. Yin W. et al. Comparative study of cnn and rnn for natural language processing // arXiv preprint arXiv:1702.01923. – 2017.
13. Connor J. T., Martin R. D., Atlas L. E. Recurrent neural networks and robust time series prediction // IEEE transactions on neural networks. – 1994. – Т. 5. – №. 2. – С. 240-254.
14. Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions //International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. – 1998. – Т. 6. – №. 02. – С. 107-116.
15. Gers F. A., Schmidhuber J., Cummins F. Learning to forget: Continual prediction with LSTM. // 9th International Conference on Artificial Neural Networks: ICANN '99. – 1999 – P. 850 – 855.
16. Olah C. Understanding LSTM Networks [Электронный ресурс] / Christopher Olah. – 2015. – Режим доступа: https://colah.github.io/posts/2015-08-Understanding-LSTMs/, свободный. - Загл. с экрана.
17. Wang J. et al. Dimensional sentiment analysis using a regional CNN-LSTM model // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). – 2016. – Т. 2. – С. 225-230.
18. A. Amidi, S. Amidi Convolutional Neural Networks cheatsheet [Электронный ресурс] Afshine Amidi, Shervine Amidi. – 2018. – Режим доступа: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks, свободный. - Загл. с экрана.
19. Britz D. Understanding convolutional neural networks for NLP [Электронный ресурс] / Denny Britz. – 2015. – Режим доступа: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/, свободный. - Загл. с экрана.
20. Kim Y. Convolutional neural networks for sentence classification // arXiv preprint arXiv:1408.5882. – 2014.
21. Wang Y. et al. A comparison of word embeddings for the biomedical natural language processing // Journal of biomedical informatics. – 2018. – Т. 87. – P. 12-20.
22. Word representations: A simple and general method for semi-supervised learning. // Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. – 2010 – P. 384–394.

23. Mikolov T. et al. Efficient estimation of word representations in vector space // arXiv preprint arXiv:1301.3781. – 2013.
24. Çano E. Text-based sentiment analysis and music emotion recognition // arXiv preprint arXiv:1810.03031. – 2018.
25. Pennington J., Socher R., Manning C. Glove: Global vectors for word representation // Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). – 2014. – P. 1532-1543.
26. Peters M. et al. Deep Contextualized Word Representations // Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). – 2018. – P. 2227-2237.
27. Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding // arXiv preprint arXiv:1810.04805. – 2018.
28. M. Eric Deep Contextualized Word Representations with ELMo [Электронный ресурс] / Mihail Eric. – 2015. – Режим доступа: https://www.mihaileric.com/posts/deep-contextualized-word-representations-elmo/, свободный. - Загл. с экрана.
29. Смирнова О. С., Петров А. И., Бабийчук Г. А. Основные методы анализа, используемые при исследовании социальных сетей // Современные информационные технологии и ИТ-образование. – 2016. – Т. 12. – №. 3-1.
30. Go A., Bhayani R., Huang L. Twitter sentiment classification using distant supervision // CS224N Project Report, Stanford. – 2009.
31. Wilson T., Wiebe J., Hoffmann P. Recognizing contextual polarity in phrase-level sentiment analysis // Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing. – 2005.
32. Mohammad S. M., Turney P. D. Crowdsourcing a word–emotion association lexicon // Computational Intelligence. – 2013. – Т. 29. – №. 3. – P. 436-465.
33. Hu M., Liu B. Mining and summarizing customer reviews // Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. – ACM, 2004. – P. 168-177.
34. Potts C. Sentiment Symposium Tutorial: Linguistic structure // Sentiment Analysis Symposium held at San Francisco. – 2011.
35. Baziotis C., Pelekis N., Doulkeridis C. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis

// Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017). – 2017. – P. 747-754.
36. Napoles C. et al. Finding good conversations online: The Yahoo News annotated comments corpus // Proceedings of the 11th Linguistic Annotation Workshop. – 2017. – P. 13-23.