Моделирование и анализ данных
2022. Том 12. № 1. С. 27–48
DOI: https://doi.org/10.17759/mda.2022120103
ISSN: 2219-3758 (печатный)
ISSN: 2311-9454 (online)

Modelling and Data Analysis
2022. Vol. 12, no. 1, pp. 27–48
DOI: https://doi.org/10.17759/mda.2022120103
ISSN: 2219-3758 (print)
ISSN: 2311-9454 (online)

◇◇◇◇◇◇◇ МЕТОДЫ ОПТИМИЗАЦИИ ◇◇◇◇◇◇◇

УДК 004.85

Выявление и классификация токсичных высказываний методами машинного обучения

Платонов Е.Н.*
Московский авиационный институт
(национальный исследовательский университет)
(ФГБОУ ВО МАИ), г. Москва, Российская Федерация
ORCID: https://orcid.org/0000-0001-8502-1350
e-mail: en.platonov@gmail.com

Руденко В.Ю.**
Московский авиационный институт
(национальный исследовательский университет)
(ФГБОУ ВО МАИ), г. Москва, Российская Федерация
ORCID: https://orcid.org/0000-0003-0010-331X
e-mail: super.ruden2011@mail.ru

Количество оставленных комментариев на платформах социальных сетей может составлять несколько миллионов в день, поэтому их владельцы заинтересованы в автоматической фильтрации контента. В данной работе рассматривается задача выявления оскорбительных высказываний в текстах. При решении задачи были рассмотрены различные метолы векторного преобразования текстов: TF-IDF, Word2Vec, GloVe и т.д. Так же были рассмотрены и представлены результаты применения классических методов классификации текста и нейросетевых методов (LSTM, CNN).

Ключевые слова: обработка текстов на естественном языке (NLP), задача классификации, градиентный бустинг, XGBoost, CatBoost, реккурентные нейронные сети, LSTM, сверточные нейронные сети.

***Платонов Евгений Николаевич,** кандидат физико-математических наук, доцент, Московский авиационный институт (национальный исследовательский университет) (ФГБОУ ВО МАИ), г. Москва, Российская Федерация, ORCID: https://orcid.org/0000-0001-8502-1350, e-mail: en.platonov@gmail.com

****Руденко Вероника Юрьевна,** студент магистратуры института «Информационные технологии и прикладная математика» Московский авиационный институт (национальный исследовательский университет), г. Москва, Российская Федерация, ORCID: https://orcid.org/0000-0003-0010-331X, e-mail: super.ruden2011@mail.ru

Для цитаты:
Платонов Е.Н., Руденко В.Ю. Выявление и классификация токсичных высказываний методами машинного обучения// Моделирование и анализ данных. 2022. Том 12. № 1. С. 27–48. DOI: https://doi.org/10.17759/mda.2022120103

1. ВВЕДЕНИЕ

Обнаружение оскорбительного (токсичного) контента контента – одна из задач обработки естественного языка.

Обработка естественного языка (NLP) – это общее направление искусственного интеллекта и математической лингвистики. Она изучает проблемы компьютерного анализа и синтеза естественных языков и представляет собой огромный спектр задач разного уровня:

- распознавание текста, синтез речи;
- морфологический анализ, канонизация;
- синтаксический разбор, токенизация предложений;
- извлечение отношений, определение языка, анализ эмоциональной окраски и т.д. [1].

Проблема обнаружения токсичного контента является актуальной, так как платформы социальных сетей обеспечивают среду, в которой люди могут свободно участвовать в дискуссиях, узнавать о тенденциях, новостях и т.д.

Для частичного решения проблемы Google и Jigsaw выпустили экспериментальное расширение Tune, позволяющее управлять показом комментариев, которые пользователи видят в сети. Tune работает с Perspective API [2], который учится помечать негативные комментарии тысяч людей, помечая миллионы постов как спам, домогательства или непристойный контент. Как только комментарий определен как токсичный, Tune может настроить видимость таких комментариев.

В работах [3,4] рассматривается аналогичная задача классификации оскорбительных комментариев. Авторы работы [3] сравнивают различные подходы глубокого обучения к решению данной задачи на двух наборах данных: комментарии на страницах обсуждений Википедии и набор данных социальной сети Twitter. Были использованы такие архитектуры нейронных сетей как: сверточная нейронная сеть (CNN), рекуррентная нейронная сети на основе Gated Recurrent Unit (GRU), двунаправленная рекуррентная сеть (Bi-GRU). Авторы проводят детальный анализ ошибок первого (False Positive) и второго рода (False Negative). Причинами появления ошибок False Negative являются: токсичность предложения без использования ругательств, риторические вопросы, сарказм и ирония. Причинами появления ошибок False Positive являются: использование нецензурных слов в ложных срабатываниях, цитаты или ссылки.

Авторы работы [4] представляют различные подходы глубокого обучения, такие как CNN, LSTM, GRU. В качестве наборов данных были использованы общедоступные наборы данных, такие как Yahoo News Annotated Comments Corpus, комментарии

на страницах обсуждений Википедии, One Million Posts Corpus. Авторы использовали не двоичную, а мультиклассовую классификацию. Больше классов требует больше данных для обучения. Обычно для англоязычных текстов доступны большие объемы обучающих данных. Тем не менее, для менее распространенных языков обучающие данные редки, а иногда маркированные данные полностью отсутствуют. Одним из способов решения этой проблемы является машинный перевод англоязычного набора данных на другой язык. Если машинный перевод хорошего качества, аннотации англоязычных комментариев также применяются к переведенным комментариям. Поэтому для подобного расширения учебных данных были использованы исследования в области трансферного обучения [5].

В статье [6] было проведено исследование аналогичной задачи для корпусов текстов на русском и украинском языках. Основная идея представленного подхода состоит в том, чтобы использовать начальный словарь оскорбительных терминов в сочетании с неконтролируемым присвоением меток (оскорбительных или не оскорбительных) комментариям в социальных сетях, а затем итеративно его расширять оскорбительными словами.

В нашей работе задача обнаружение оскорбительного контента рассматривается как задача бинарной классификации.

2. ВЕКТОРНОЕ ПРЕДСТАВЛЕНИЕ ТЕКСТА

Векторное представление текста (word embedding) – это собирательное название для набора методов моделирования и обработки естественного языка (NLP), где слова или фразы из словаря отображаются в пространство действительных чисел.

Векторная модель – представление текстов вещественными векторами из одного общего для всей коллекции текстов векторного пространства. с фиксированной размерностью Размерность пространства равна количеству различных слов во всей текстовой коллекции.

Вектор образуется упорядочением весов всех слов, включая те, которых нет в конкретном тексте. Размерность этого вектора, как и размерность пространства, равна количеству различных слов во всей коллекции, и является одинаковой для всех текстов коллекции.

Формально:

$$d_j = (w_{j1}, w_{j2}, \dots, w_{jn})$$

где d_j – векторное представление -го текста, w_{ji} – вес i-го слова в j-м тексте, n – общее количество различных слов во всех текстах коллекции.

Для полного определения векторной модели необходимо указать, каким именно образом будет отыскиваться вес слова в документе [7].

Векторное представление текста является основным способом решением задач информационного поиска: поиск документа по запросу, классификация документов, кластеризация документов и т.д.

Рассмотрим несколько методов векторного представления слов.

TF-IDF

В данном методе текст сводится к вектору, где каждая его компонента представляет собой слово, а значением данной компоненты является число раз, которое это слово используется в тексте.

Преобразование TF-IDF – используется для корректировки значений вектора в соответствии с числом документов, использующих слово. Слова, встречающиеся во многих документах, могут быть менее значимыми, чем слова, встречающиеся реже. TF-IDF уменьшает значение данного слова пропорционально количеству документов, в которых оно появляется.

TF или частота слова – это отношение количества вхождения конкретного термина к суммарному набору слов в исследуемом тексте или же документе.

IDF или обратная частота документа – это инверсия частотности, с которой определенное слово фигурирует в коллекции текстов [8].

Word2Vec

Word2Vec – технология от Google [9], использующаяся для статистического анализа больших массивов текстовой информации. Она собирает статистику по совместному появлению слов в фразах, после этого с помощью нейронных сетей решает задачу уменьшения размерности и в итоге выдает компактные векторные представления слов, достаточно полно отражающие отношения этих слов в обрабатываемых текстах.

Данная модель предсказывает вероятность слова по его окружению – контексту. То есть формируются такие вектора слов, чтобы вероятность, присваиваемая моделью слову, была близка к вероятности встретить слово в этом окружении в реальном тексте [10].

FastText

FastText – это библиотека для изучения встраивания слов и классификации текста, созданная исследовательской лабораторией AI в Facebook. Модель позволяет создать алгоритм обучения без контроля или обучения для получения векторных представлений для слов.

Для эффективной обработки массивов данных с большим количеством различных категорий FastText использует иерархический классификатор, который организует категории в древовидную структуру вместо плоской.

FastText является расширением Word2Vec. Для модели векторных представлений слов используется *skip-gram* с негативным сэмплированием.

Негативное сэмплирование – это способ создать для обучения векторной модели отрицательные примеры, то есть показать ей пары слов, которые не являются соседями по контексту (например, «пушистый котик» → «пушистый утюг»). Всего подбирается от 3 до 20 случайных слов. Такой случайный подбор нескольких примеров не требует много компьютерного времени и позволяет ускорить работу FastText [11].

GloVe

GloVe (Global Vectors) – модель, предложенная лабораторией компьютерной лингвистики Стенфордского университета. Данный алгоритм сочетает в себе черты

SVD разложения и Word2Vec. Метод GloVe основан на идее выведения семантических отношений между словами из матрицы совпадений.

По входному словарю V, строится частотная матрица совпадений V x V, где элемент матрицы X_{ij} обозначает, сколько раз слово i встречалось со словом j.

	the	cat	sat	on	mat
the	0	1	0	1	1
cat	1	0	1	0	0
sat	0	1	0	1	0
on	1	0	1	0	0
mat	1	0	0	0	0

Рис. 1. Пример матрицы совпадений

Более подробно работа GloVe описана в статье [12].

Paragram

Paraphrastic sentence embeddings (Paragram) — универсальный эмбеддинг, основанный на комбинировании вложения слов для получения вложений предложений, удовлетворяющих тому свойству, что вектора предложений, являющиеся парафразами друг друга, расположены рядом друг с другом в векторном пространстве.

Парафраз – выражение, являющееся описательной передачей смысла другого выражения или слова.

Комбинирование проводится контролируемым образом на основе наблюдения из базы данных известных парафразов.

Цель алгоритма состоит в том, чтобы встроить последовательности в низкоразмерное пространство таким образом, чтобы косинусное сходство в пространстве соответствовало силе связи парафразирования между последовательностями [13].

3. МЕТОДЫ

В работе используются методы такие как логистическая регрессия, градиентный бустинг (XGBoost, CatBoost), нейронная сеть долговременной краткосрочной памяти (LSTM) и сверточная нейронная сеть (CNN).

Градиентный бустинг

Градиентный бустинг — техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений.

Обучение ансамбля проводится последовательно. На каждой итерации вычисляются отклонения предсказаний уже обученного ансамбля на обучающей выборке.

Следующая модель, которая будет добавлена в ансамбль будет стараться уменьшить эти отклонения. Таким образом, добавив предсказания нового дерева к предсказаниям обученного ансамбля происходит уменьшение среднего отклонения модели. Новые деревья добавляются в ансамбль до тех пор, пока ошибка уменьшается, либо пока не выполняется одно из правил «ранней остановки».

Метод градиентного бустинга обладает высокой гибкостью для решения задач классификации.

В данной работе будет использовано две реализации градиентного бустинга:

- XGBoost (eXtreme Gradient Boosting);

В отличие от стандартного градиентного бустинга в методе XGBoost построение деревьев основано на параллелизации, а в качестве критерия остановки разбиения дерева используется параметр максимальной глубины. Для нахождения оптимальных точек разделения используется метод взвешенных квантилей [14]. Так же алгоритм содержит возможность добавления L1 или L2 регуляризацию.

Еще одним значительным улучшением XGBoost является возможность упрощенной работы с разреженными матрицами.

Алгоритм работы XGBoost подробно описан в статье [14].

- CatBoost (Categorical Boosting).

Библиотека CatBoost (Categorical Boosting) – метод машинного обучения, основанный на градиентном бустинге, разработанный инженерами Яндекса. Главное преимущество которой заключается в том, что она одинаково хорошо работает как с числовыми признаками, так и с категориальными.

CatBoost основан на двоичных деревьях решений с градиентным бустингом. Во время обучения набор деревьев решений строится последовательно. Каждое последующее дерево строится с меньшими потерями по сравнению с предыдущими деревьями. Более подробно алгоритм и преимущество использования открытой библиотеки CatBoost описано в работе [15]. В статье также приведены результаты сравнения метода CatBoost с другими методами, использующими градиентный бустинг.

LSTM

Сеть долговременной краткосрочной памяти LSTM (Long Short-Term Memory) – частный случай рекуррентной нейронной сети (RNN).

RNN представляет собой сети с петлями в них, что позволяет хранить информацию о том, что было в предшествующем предложении. RNN используют обратное распространение ошибки в процессе обучения для обновления весов сети на каждом уровне.

Сеть долговременной кратковременной памяти была изобретена с целью решения проблем исчезающих и взрывающихся градиентов. Ключевым моментом в разработке LSTM было включение нелинейных, зависящих от данных элементов управления в ячейку RNN, которые могут быть обучены для обеспечения того, чтобы градиент целевой функции по отношению к сигналу состояния не исчезал [16].

Рассмотрим модуль LSTM, называемый блоком памяти, на основе работ [17, 18].

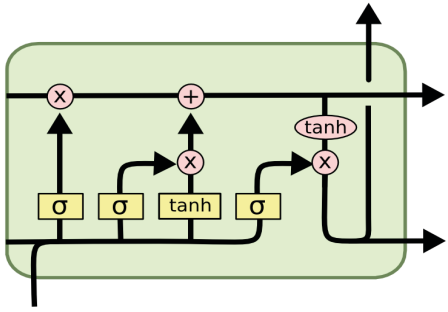

Рис. 2. Блок памяти

Входные затворы (gates), которые представляют собой простые сигмоидальные пороговые блоки с диапазоном функций активации [0, 1], управляют сигналами от сети к ячейке памяти, соответствующим образом масштабируя их; когда затвор закрыт, активация близка к нулю. Сигмоидальные затворы состоят из слоя сигмовидной нейронной сети и операции точечного умножения. Кроме того, они могут научиться защищать содержимое от помех со стороны неуместных сигналов.

Выходные затворы могут научиться управлять доступом к содержимому ячейки памяти, которое защищает другие ячейки памяти от помех. Выходной затвор решает какую информацию от предыдущих шагов необходимо сохранить. Результат будет являться отфильтрованным состоянием ячейки. Сначала текущий ввод проходит через сигмоидальный функцию, затем пропускаем состояние ячейки через функцию гиперболического тангенса (*tanh*). Вывод функции *tanh* перемножается на вывод сигмоидальной функции.

CNN

Сверточная нейронная сеть (Convolutional Neural Network, CNN) – архитектура нейронных сетей, как аналог зрительной коры головного мозга для распознавания изображений.

Структура сети – однонаправленная, принципиально многослойная. Для обучения используются стандартные методы, чаще всего метод обратного распространения ошибки.

Классическая архитектура CNN обычно имеет 3 типа слоев:

1. Сверточный слой.
2. Слой субдискретизации (pooling).
3. Полносвязный слой.

Сверточный слой получает в качестве входных данных трехмерную матрицу размера $[n_1, m_1, d_1]$.

Ядро (фильтр) сверточного слоя (сверточная матрица) представляет собой матрицу с размерами $[n_2, m_2, d_1]$, т.е. глубина d_1 ввода и одного ядра одинаковы. Для каждого сверточного слоя имеется несколько ядер, уложенных друг на друга, что образует матрицу с размерами $[n_2, m_2, d_2]$, где d_2 – количество ядер. Для каждого ядра есть соответствующее смещение, которое является скалярной величиной.

Выходом сверточного слоя является матрица размерностью $[n_3, m_3, d_2]$, глубина вывода равна количеству ядер.

Основная цель слоя субдискретизации состоит в том, чтобы уменьшить количество параметров входа. Входные данные для данного слоя являются тензорными.

Полносвязный слой – это просто нейронная сеть с обратной связью. Полносвязные слои образуют последние несколько слоев в сети. После прохождения через них последний слой использует функцию активации softmax, которая используется для получения вероятности того, что входные данные относятся к определенному классу [19].

Было показано, что сверточные нейронные сети, первоначально изобретенные для компьютерного зрения, обеспечивают высокую производительность при решении задач классификации текста [20].

В настоящее время предполагается, что CNN классифицирует текст, выполняя следующие шаги.

1. В качестве детекторов n-граммы используются одномерные сверточные матрицы, каждая из которых специализируется на тесно связанном семействе n-граммов. Ядра сверточного слоя не являются однородными, т. е. одно ядро может и часто обнаруживает несколько явно разных семейств n-грамм. Ядра также обнаруживают отрицательные элементы в n-граммах.
2. Слой субдискретизации с функцией максимума (Max Pooling) с течением времени извлекает соответствующие n-граммы для принятия решения. Max Pooling так же вызывает пороговое поведение, и значения ниже заданного порога игнорируются при прогнозировании.
3. Остальная часть сети классифицирует текст на основе этой информации.

4. РЕЗУЛЬТАТЫ И ОБСУЖДЕНИЕ

Рассмотрим задачу бинарной классификации текстов на примере конкурса от компании Quora с платформы Kaggle), в котором необходимо предсказать является ли вопрос, заданный на платформе *Quora* «искренним» – нейтральным или «неискренним» – токсичным [21].

Файл данных включает в себя 80810 неискренних (токсичных) записей и 1225312 нейтральных записей. Следовательно, при использовании методов решения задачи классификации следует учитывать тот факт, что работа будет вестись с несбалансированным набором данных.

Приведем часть данных в виде таблицы 1.

Таблица 1

Данные из набора Quora

	question text	**target**
0	How did Quebec nationalists see their province as a nation in the 1960s?	0
1	Do you have an adopted dog, how would you encourage people to adopt and not shop?	0
22	Has the United States become the largest dictatorship in the world?	1

	question text	**target**
…	…	…
1306093	How is it to have intimate relation with your cousin?	1
1306112	Are you ashamed of being an Indian?	1

Качество решения задачи классификации будем оценивать с помощью матрицы ошибок.

Таблица 2

Матрица ошибок

	$a = 0$	$a = 1$
$y = 0$	TP	FP
$y = 1$	FN	TN

Два класса делятся на положительный (обычно метка 1) и отрицательный (обычно метка 0). Объекты, которые алгоритм относит к положительному классу, называются положительными (Positive), те из них, которые на самом деле принадлежат к этому классу – истинно положительными (True Positive), остальные – ложно положительными (False Positive). Аналогичная терминология есть для отрицательного (Negative) класса. В табл. 2 использованы сокращения: TP = true positive, TN = true negative, FP = false positive, TN = true negative.

При построении матрицы ошибок также используются нормированные значения.

Таблица 3

Нормированная матрица ошибок

	$a = 0$	$a = 1$
$y = 0$	$\frac{TP}{TP + FN}$	$\frac{FN}{FN + TP}$
$y = 1$	$\frac{FP}{FP + TN}$	$\frac{TN}{TN + FP}$

Часто результат работы алгоритма на фиксированной тестовой выборке визуализируют с помощью ROC-кривой (receiver operating characteristic) [22], а качество оценивают как площадь под этой кривой – AUC (area under the curve). AUC ROC равен доле пар объектов вида (объект класса 1, объект класса 0), которые алгоритм верно упорядочил, т.е. первый объект идёт в упорядоченном списке раньше.

Применение классических методов

Рассмотрим решение поставленной задачи с помощью базовых алгоритмов классификации: логистической регрессии [23], XGBoost и CatBoost.

В этом разделе в качестве эмбединга будем использовать TF-IDF с размерностью выходного вектора равной 300.

В данных присутствует сильный дисбаланс классов (большой перевес в сторону нейтральных записей), это значит, что без настройки модели будут хорошо предсказывать нейтральный записи и плохо токсичные, при этом значении метрики качества будет высоким. Суть задачи же заключается как раз в поиске токсичных записей.

Для работы с несбалансированными данными есть несколько подходов. Одним из них является указание весов для каждого класса. Для каждой модели значение весов указывается по-разному.

Для инициализации метода логистической регрессии был использован параметр $class\_weight = 'balanced'$. Метол логистической регрессии при данном параметре использует значения целевой переменной y для автоматической регулировки весов, обратно пропорциональных частотам классов во входных данных.

Построим ROC-кривую и матрицу ошибок для метода логистической регрессии.

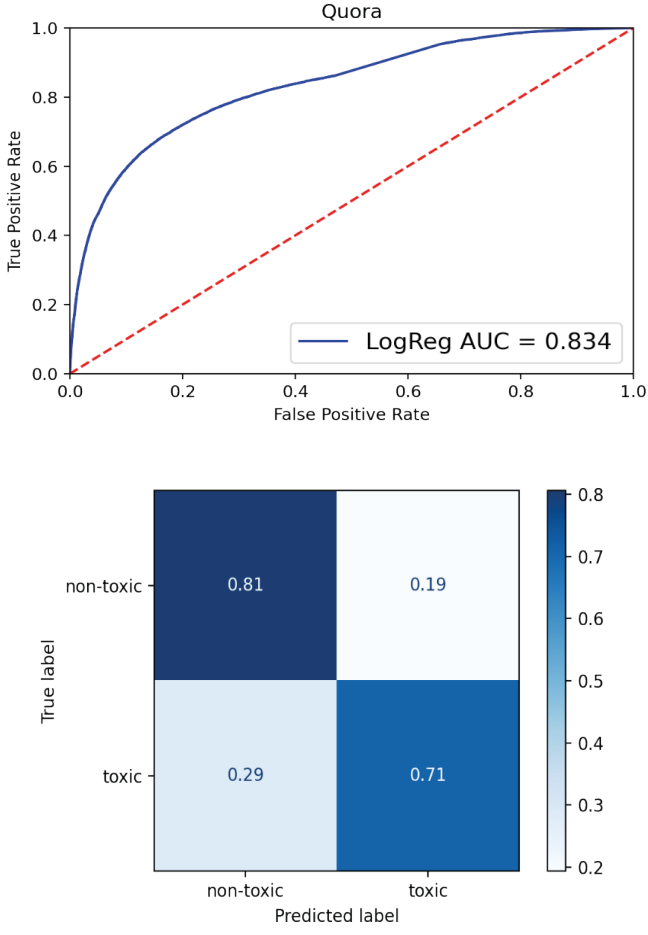

Рис. 4. ROC-кривая и матрица ошибок для логистической регрессии

Выведем топ-10 весов слов для каждого класса с помощью библиотеки для интерпретации результатов eli5 [24].

y=toxic top features

Weight[?]	Feature
+4.975	muslims
+4.437	indians
+4.271	americans
+4.071	trump
+3.683	muslim
+3.524	women
+3.353	girls
+3.086	white
+2.930	hate
+2.863	sex
… 150 more positive …	
… 131 more negative …	
-2.279	best
-2.384	rank
-2.407	energy
-2.429	app
-2.516	prepare
-2.683	tips
-2.762	marketing
-2.978	engineer
-3.084	marks
-3.101	engineering

Рис. 5. Топ-10 весов слов для каждого класса

Зеленым выделены веса слов, по которым метод логистической регрессии относит текст к классу токсичный или неискренний, красным соответственно веса слов для класса нейтральных записей. Например, если в вопросе присутствует слово «muslim», то он с большей вероятностью будет отнесен к классу токсичных вопросов.

При инициализации метода CatBoost были переданы веса классов: $class_{weights} = [0.06, 0.94]$. Веса классов для данной модели были вычислены как отношение:

$$count_0 / n; count_1 / n,$$

где $count_0$ – количество нейтральных записей в обучающей выборке, $count_1$ – количество оскорбительных записей в обучающей выборке, n – размер обучающей выборки.

Для инициализации метода XGBoost был рассчитано отношение записей на обучающем наборе нейтрального класса к классу токсичных вопросов: $estimate = 15.160$.

В метод был передан параметр $scale\_pos\_weight = (estimate + 10) = 25.160$. Добавка $+10$ была подобрана эмпирически. Параметр отвечает за балансировку положительных и отрицательных весов.

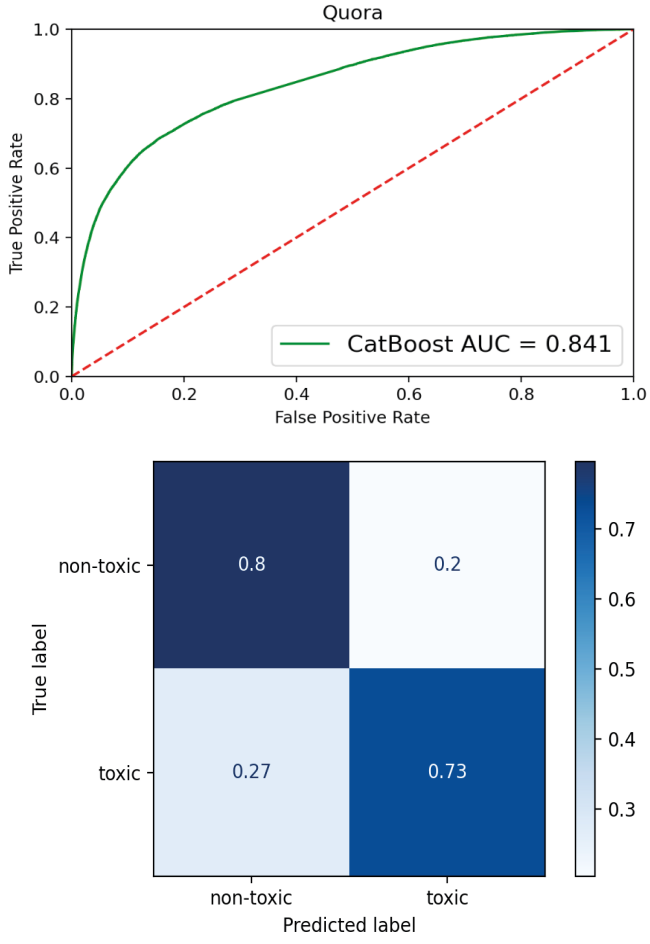

Рис. 6. ROC-кривая и матрица ошибок для CatBoost

В примере с логистической регрессией были показаны топ 10 весов для каждого класса, которые вносят наибольший вклад при классификации текстов. Для дальнейших примеров данную таблицу выводить не будем и будем рассматривать интерпретацию отдельного текста из коллекции с помощью алгоритма LIME для модели XGBoost.

LIME (Local Interpretable Model Agnostic Explanations) – алгоритм, который может объяснить предсказания классификатора или регрессора, локально аппроксимируя его интерпретируемой моделью [25].

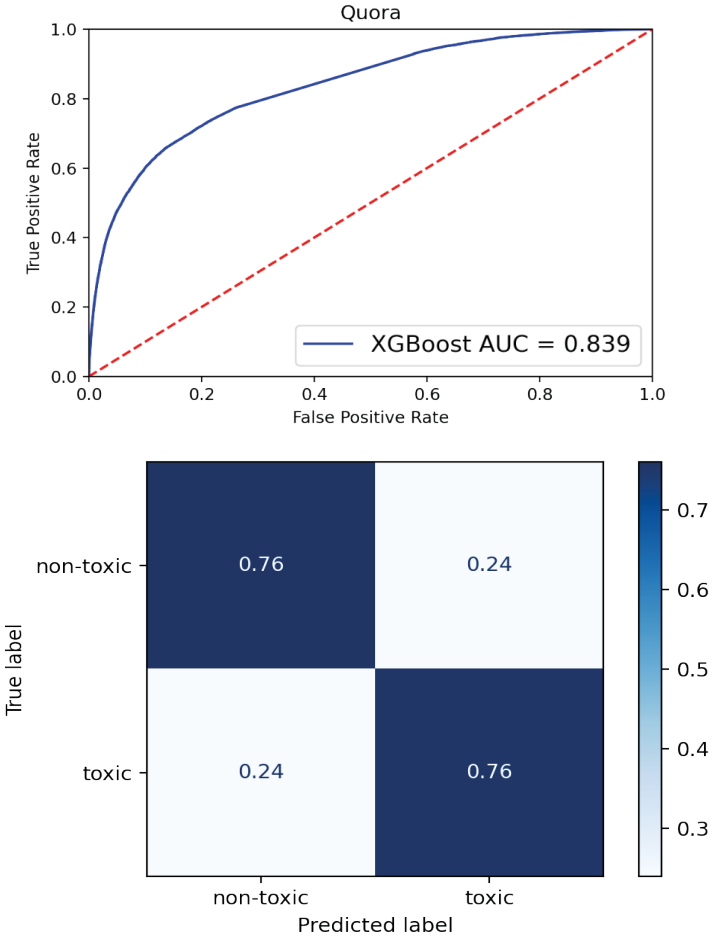

Рис. 7. ROC-кривая и матрица ошибок для XGBoost

Благодаря весу слова «working» текст был определен как нейтральный. Веса всех слов кроме «working» достаточно малы и поэтому на представленном рисунке отображаются со значениями 0.00. Алгоритм LIME позволяет отдельно вывести все веса слов и объяснить их значения для каждого класса. Так как значение данного класса равно 0 выведем веса слов для него:

$$Explanation\ for\ class\ 0$$
$$('working', 0.0732)$$
$$('the', 8.8608e^{-6})$$
$$('of', 8.8331e^{-6})$$
$$('stands', 8.0563e^{-6})$$

$('What', 6.6930e^{-6})$
$('potentiometer', 6.4858e^{-6})$
$('which', 5.9871e^{-6})$
$('basics', 4.7359e^{-6})$
$('upon', 4.3065e^{-6})$
$('are', 3.5078e^{-6})$

Так же получим результаты для Logistic Regression, XGBoost и CatBoost с применением эмбедингов Word2Vec и FastText.

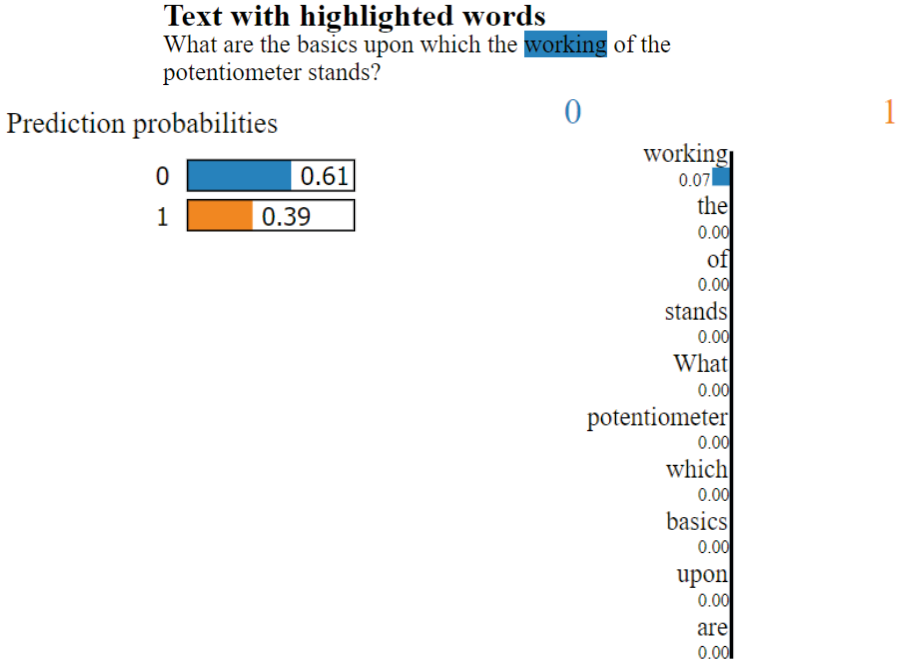

Рис. 8. Интерпретация результатов с помощью LIME

Таблица 4

Сводная таблица точности AUC

	Word2Vec		TF-IDF		FastTex	
	ROC-AUC	**Time**	**ROC-AUC**	**Time**	**ROC-AUC**	**Time**
LogReg	0.668	0.81	0.835	0.17	0.665	0.76
CatBoost	0.761	10.15	0.839	12.08	0.759	9.59
XGBoost	0.751	30.24	0.841	0.38	-	-

Из полученных результатов видно, что при использовании TF-IDF стандартные модели показывают более высокое значение метрики качества.

Применение методов глубокого обучения

Чтобы сократить время обучения моделей из этого раздела и подготовки текста для векторного представления был использован предобученный Embedding слой.

Предобученные эмбеддинги обычно представляют собой словари, где слову ставится в соответствие вектор. Такие словари получаются следующим образом: эмбеддинг обучается на больших корпусах текстов с использованием специальных моделей, а затем выгружается в виде словаря. Таким образом можно получить качественный эмбеддинг без необходимости его повторного обучения.

Зададим размер словаря передаваемый в Embedding слой равным 50000 уникальных слов. Для покрытия большинства примеров достаточно взять длину предложения равную 20 словам (рис. 9).

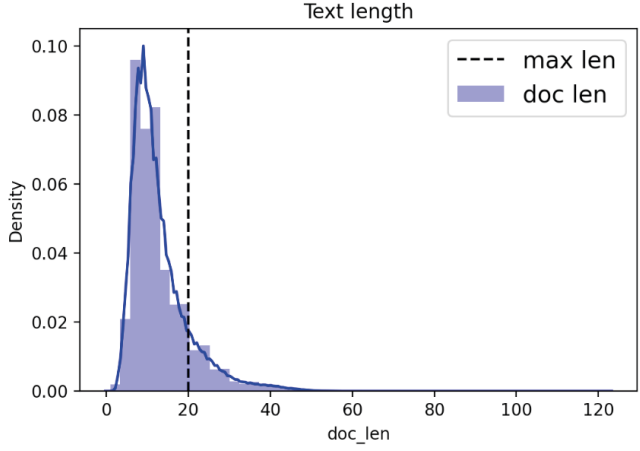

Рис. 9. Гистограмма длин предложений в обучающей выборке

В аргумент weights Embedding слоя передается эмбеддинг матрица, составленная из четырех предобученных эмбеддингов: GloVe [26], FastText [27], Paragram [28] и Word2Vec [29]. Использование ансамбля предобученных эмбеддингов позволит моделям более точно классифицировать тексты.

Для каждого из представленных эмбедингов выходной вектор имеет размерность 300, как и в случае с TF-IDF из прошлого раздела. Тогда выходная размерность Embedding слоя равна $300 \times 4 = 1200$.

Для моделей глубокого обучения вычислим веса классов как:

$$n\_samples \,/\left(n_{classes} * bincount\left(y\right)\right),$$

где $n\_samples$ – размер выборки, $n_{classes}$ – количество классов (в нашем случае 2), $bincount\left(y\right)$ – функция, подсчитывающая количество вхождений каждого класса в выборку.

В таком случае веса классов равны: $\left\{0:0.533, 1:8.062\right\}$.

Рассмотрим решения получаемы с использованием LSTM и CNN.

Сеть LSTM была обучена с batch_size = 1024 и количеством эпох равном 5. Построим ROC-кривую и матрицу ошибок.

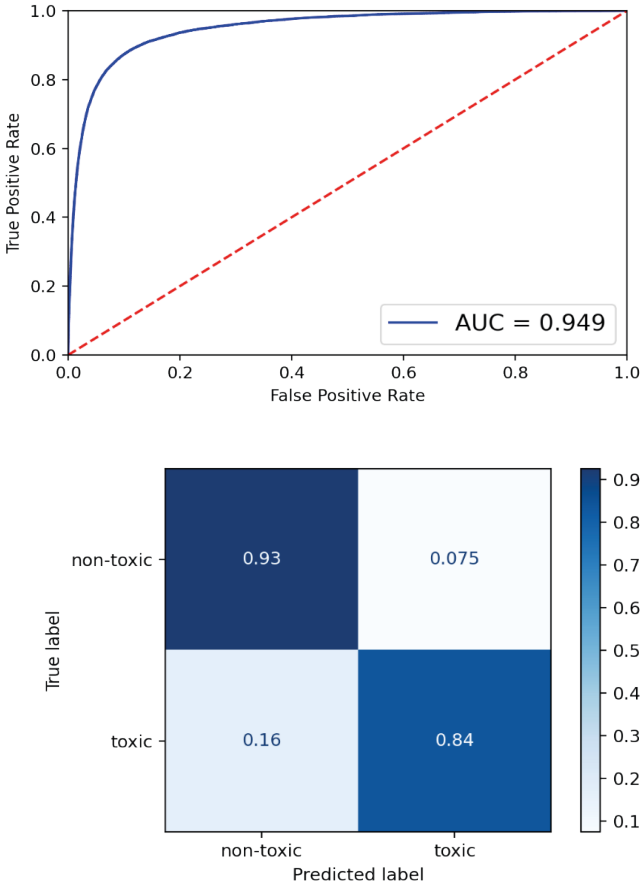

Рис. 10. ROC-кривая и матрица ошибок для LSTM

Сеть CNN была обучена с batch_size = 512 и двумя эпохами обучения. Построим ROC-кривую и матрицу ошибок.

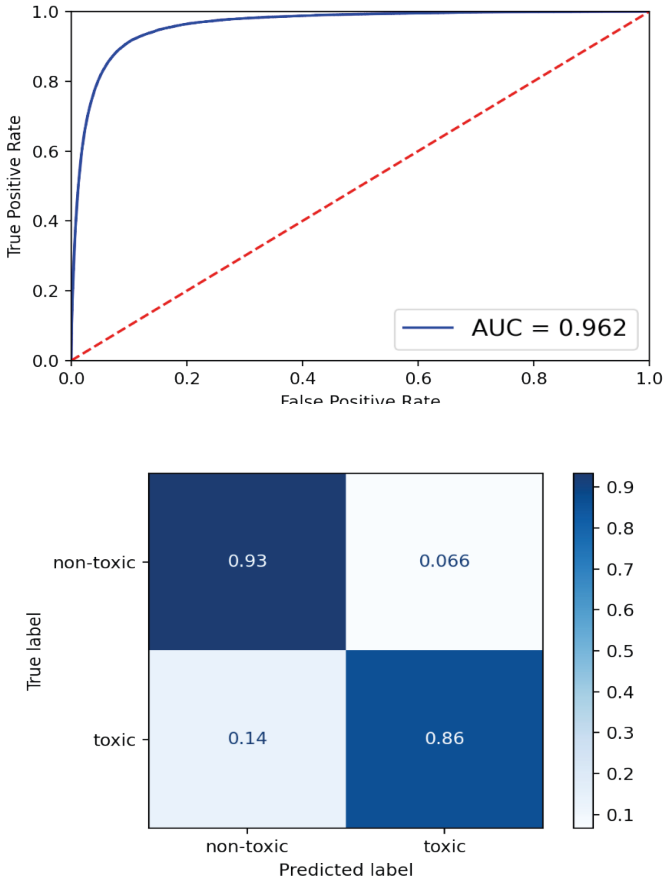

Рис. 11. ROC-кривая и матрица ошибок для CNN

Проинтерпретируем результаты архитектуры LSTM для того же предложения с помощью алгоритма LIME.

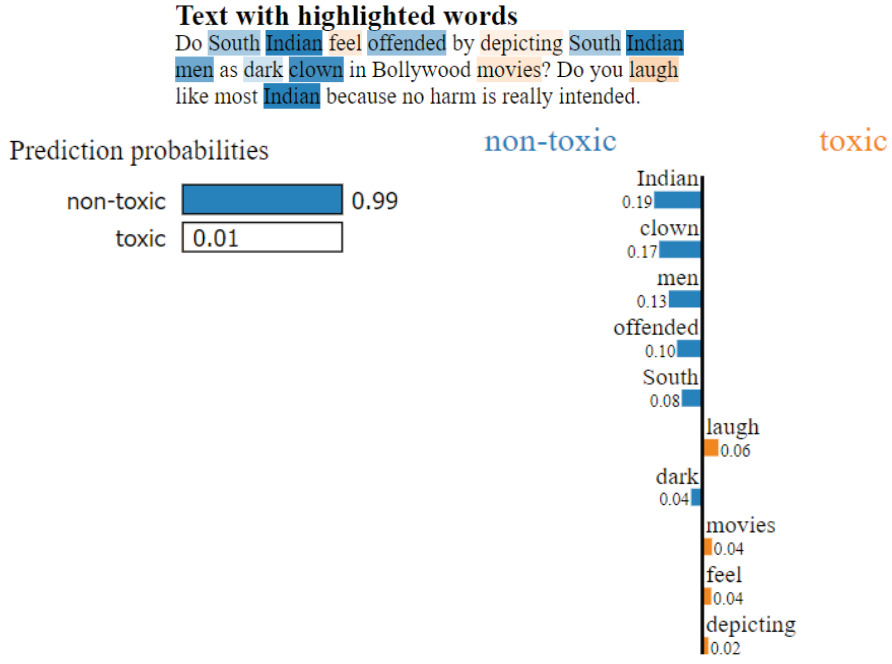

Рис.12. Интерпретация результатов с помощью LIME

На рис. 12 можно увидеть благодаря весам каких слов данный текст был отмечен как нейтральный.

4. ЗАКЛЮЧЕНИЕ

Сравним результаты, полученные с помощью моделей глубоко обучения, с классическими моделями решения задачи классификации.

Таблица 5

Сводная таблица точности AUC и элементов матрицы

Применяемые методы	ROC-AUC	balanced accuracy
LogReg	0.835	0.760
CatBoost	0.839	0.765
XGBoost	0.841	0.760
LSTM	0.949	0.875
CNN	0.962	0.895

Balanced accuracy – показатель, который можно использовать при оценке того, насколько хорош, бинарный классификатор:

$$Balanced\ accuracy = \frac{1}{2}\left(\frac{TP}{TP+FP}+\frac{TN}{TN+FP}\right).$$

Видно, что CNN превосходит как классические модели, так и LSTM по значению метрики качества и по значению balanced accuracy.

Результаты для обеих моделей LSTM и CNN были получены с использованием малого числа эпох, когда как для количества тренируемых параметров, значение которых достигает нескольких миллионов, этого явно недостаточно. Так же из-за факта обучения на малом количестве эпох вытекает необходимость исследования на предмет возникновения эффекта переобучения моделей.

При дальнейшей работе с бинарной классификацией текстов имеет смысл обратить внимание на такие модели как Bi-LSTM, GRU и Bi-GRU, на применение методов ансамблирования и трансферного обучения (модели BERT, ELMo и т.д.).

Литература

1. *Риз Р.* Обработка естественного языка на Java. ДМК-Пресс. 2016. 264 с.
2. Perspective API. URL: https://www.perspectiveapi.com
3. *van Aken B., Risch J., Krestel R., Löser A.* Challenges for toxic comment classification: An in-depth error analysis. 2018, arXiv:1809.07572.
4. *Risch J., Krestel R.* Toxic Comment Detection in Online Discussions. Deep Learning-Based Approaches for Sentiment Analysis. Springer, Singapore, 2020. P. 85–109.
5. *Weiss K., Khoshgoftaar T.M., Wang D.* A survey of transfer learning // Big Data, 3: 9. 2016. https://doi.org/10.1186/s40537–016–0043–6
6. *Andrusyak B., Rimel M., Kern R.* Detection of Abusive Speech for Mixed Sociolects of Russian and Ukrainian Languages // RASLAN. – 2018. – P. 77–84.
7. *Li Y., Yang T.* Word Embedding for Understanding Natural Language: A Survey. In: Srinivasan S. (eds) Guide to Big Data Applications. Studies in Big Data, vol 26. Springer, Cham. https://doi.org/10.1007/978–3-319–53817–4_4
8. *Liu C.* et al. Research of text classification based on improved TF-IDF algorithm // IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE). 2018 P. 218–222.
9. word2vec // URL: https://code.google.com/archive/p/word2vec/
10. *Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean.* Efficient Estimation of Word Representations in Vector Space // Proceedings of Workshop at ICLR, 2013.
11. *Bojanowski P*, et al. Enriching word vectors with subword information // Transactions of the Association for Computational Linguistics. 2017. V. 5. P. 135–146.
12. *Pennington J., Socher R., Manning C.D. G*love: Global vectors for word representation // Proceedings of the conference on empirical methods in natural language processing (EMNLP). 2014. P. 1532–1543.
13. *Wieting J.* et al. From paraphrase database to compositional paraphrase model and back // Transactions of the Association for Computational Linguistics. 2015. V. 3. P. 345–358.
14. *Chen T., Guestrin C.* Xgboost: A scalable tree boosting system // Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016. P. 785–794.
15. *Dorogush A.V., Ershov V., Gulin A*. CatBoost: gradient boosting with categorical features support // arXiv preprint arXiv:1810.11363. 2018.
16. *Sepp Hochreiter and Jürgen Schmidhuber.* Long Short-Term memory // Neural computation, V. 9(8). P. 1735–1780, 1997.
17. *Staudemeyer R.C., Morris E.R.* Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks // arXiv preprint arXiv:1909.09586. 2019. URL:https://arxiv.org/pdf/1909.09586.pdf
18. Understanding LSTM Networks URL:https://colah.github.io/posts/2015–08-Understanding-LSTMs/
19. Convolutional Neural Network. An Introduction to Convolutional Neural Networks. URL: https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05

20. *Bai S., Kolter J.Z., Koltun V.* An empirical evaluation of generic convolutional and recurrent networks for sequence modeling // CoRR, abs/1803.01271. 2018. http://arxiv.org/abs/1803.01271
21. Quora Insincere Questions Classification. URL: https://www.kaggle.com/c/quora-insincere-questions-classification/data
22. *T. Fawcett.* An introduction to ROC analysis // Pattern Recognition Letters, V. 27. 2006. P. 861–874.
23. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning. Springer-Verlag, New York. 2017.
24. Eli5 Documentation. URL: https://eli5.readthedocs.io/en/latest/
25. *Tulio Ribeiro M., Singh S., Guestrin C.* Why Should I Trust You? Explaining the Predictions of Any Classifier // KDD'16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 1135–1144.
26. glove.840B.300d – pre-trained word vectors GloVe. URL: https://nlp.stanford.edu/projects/glove/
27. wiki-news-300d-1M – pre-trained word vectors trained using fastText. URL: https://fasttext.cc/docs/en/english-vectors.html
28. paragram_300_sl999 – New Paragram-SL999 300 dimensional embeddings tuned on SimLex999 dataset. URL: https://www.kaggle.com/ranik40/paragram-300-sl999
29. GoogleNews-vectors-negative300 – pre-trained word vectors trained using Word2Vec. URL: https://code.google.com/archive/p/word2vec/

Identification and Classification of Toxic Statements by Machine Learning Methods

Evgeniy N. Platonov*
Moscow Aviation Institute (National Research University) Moscow, Russia
ORCID: https://orcid.org/0000-0001-8502-1350
e-mail: en.platonov@gmail.com

Veronika Y. Rudenko**
Moscow Aviation Institute (National Research University) Moscow, Russia
ORCID: https://orcid.org/0000-0003-0010-331X
e-mail: super.ruden2011@mail.ru

The number of comments left on social media platforms can reach several million per day, so their owners are interested in automatic content filtering. In this paper, the task of identifying offensive statements in texts is considered. When solving the problem, various methods of vector text conversion were considered: TF-IDF, Word2Vec, Glove, etc. The results of the application of classical text classification methods and neural network methods (LSTM, CNN) were also considered and presented.

Keywords: Natural Language Processing (NLP), Classification, Gradient boosting, XGBoost, CatBoost, Recurrent Neural Network, LSTM, Convolutional Neural Network.

For citation:
Platonov E.N., Rudenko V.Y. Identification and Classification of Toxic Statements by Machine Learning Methods. *Modelirovanie i analiz dannykh = Modelling and Data Analysis,* 2022. Vol. 12, no. 1, pp. 27–48. DOI: https://doi.org/10.17759/mda.2022120103 (In Russ., abstr. in Engl.).

References

1. *Riz R*. Natural language processing in Java. DMK-Press. 2016. 264 p.
2. Perspective API. URL: https://www.perspectiveapi.com
3. *van Aken B., Risch J., Krestel R., Löser A.* Challenges for toxic comment classification: An in-depth error analysis. 2018, arXiv:1809.07572.
4. *Risch J., Krestel R.* Toxic Comment Detection in Online Discussions. Deep Learning-Based Approaches for Sentiment Analysis. Springer, Singapore, 2020. P. 85–109.
5. *Weiss K., Khoshgoftaar T.M., Wang D.* A survey of transfer learning // Big Data, 3: 9. 2016. https://doi.org/10.1186/s40537–016–0043–6
6. *Andrusyak B., Rimel M., Kern R.* Detection of Abusive Speech for Mixed Sociolects of Russian and Ukrainian Languages // RASLAN. – 2018. – P. 77–84.

***Evgeniy N. Platonov,** PhD (Physics and Mathematics), Assistant Professor, Moscow Aviation Institute (National Research University), Moscow, Russia, ORCID: https://orcid.org/0000-0001-8502-1350, e-mail: en.platonov@gmail.com

****Veronika Y. Rudenko,** Student of the Institute of Information Technologies and Applied Mathematics, Moscow Aviation Institute (National Research University), Moscow, Russia, ORCID: https://orcid.org/0000-0003-0010-331X, e-mail: super.ruden2011@mail.ru

7. *Li Y., Yang T.* Word Embedding for Understanding Natural Language: A Survey. In: Srinivasan S. (eds) Guide to Big Data Applications. Studies in Big Data, vol 26. Springer, Cham. https://doi.org/10.1007/978–3-319–53817–4_4
8. *Liu C.* et al. Research of text classification based on improved TF-IDF algorithm // IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE). 2018 P. 218–222.
9. word2vec // URL: https://code.google.com/archive/p/word2vec/
10. *Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean.* Efficient Estimation of Word Representations in Vector Space // Proceedings of Workshop at ICLR, 2013.
11. *Bojanowski P*, et al. Enriching word vectors with subword information // Transactions of the Association for Computational Linguistics. 2017. V. 5. P. 135–146.
12. *Pennington J., Socher R., Manning C.D. G*love: Global vectors for word representation // Proceedings of the conference on empirical methods in natural language processing (EMNLP). 2014. P. 1532–1543.
13. *Wieting J.* et al. From paraphrase database to compositional paraphrase model and back // Transactions of the Association for Computational Linguistics. 2015. V. 3. P. 345–358.
14. *Chen T., Guestrin C.* Xgboost: A scalable tree boosting system // Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016. P. 785–794.
15. *Dorogush A.V., Ershov V., Gulin A.* CatBoost: gradient boosting with categorical features support // arXiv preprint arXiv:1810.11363. 2018.
16. *Sepp Hochreiter and Jürgen Schmidhuber.* Long Short-Term memory // Neural computation, V. 9(8). P. 1735–1780, 1997.
17. *Staudemeyer R.C., Morris E.R.* Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks // arXiv preprint arXiv:1909.09586. 2019. URL:https://arxiv.org/pdf/1909.09586.pdf
18. Understanding LSTM Networks URL:https://colah.github.io/posts/2015–08-Understanding-LSTMs/
19. Convolutional Neural Network. An Introduction to Convolutional Neural Networks. URL: https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05
20. *Bai S., Kolter J.Z., Koltun V.* An empirical evaluation of generic convolutional and recurrent networks for sequence modeling // CoRR, abs/1803.01271. 2018. http://arxiv.org/abs/1803.01271
21. Quora Insincere Questions Classification. URL: https://www.kaggle.com/c/quora-insincere-questions-classification/data
22. *T. Fawcett.* An introduction to ROC analysis // Pattern Recognition Letters, V. 27. 2006. P. 861–874.
23. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning. Springer-Verlag, New York. 2017.
24. Eli5 Documentation. URL: https://eli5.readthedocs.io/en/latest/
25. *Tulio Ribeiro M., Singh S., Guestrin C.* Why Should I Trust You? Explaining the Predictions of Any Classifier // KDD'16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 1135–1144.
26. glove.840B.300d – pre-trained word vectors GloVe. URL: https://nlp.stanford.edu/projects/glove/
27. wiki-news-300d-1M – pre-trained word vectors trained using fastText. URL: https://fasttext.cc/docs/en/english-vectors.html
28. paragram_300_sl999 – New Paragram-SL999 300 dimensional embeddings tuned on SimLex999 dataset. URL: https://www.kaggle.com/ranik40/paragram-300-sl999
29. GoogleNews-vectors-negative300 – pre-trained word vectors trained using Word2Vec. URL: https://code.google.com/archive/p/word2vec/

Получена 18.01.2022
Принята в печать 31.01.2022

Received 18.01.2022
Accepted 31.01.2022