КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ
COMPUTER SIMULATION HISTORY

УДК 004.853
DOI 10.52575/2687-0932-2021-48-1-100-115

Реализация классификатора групп в социальных сетях с помощью рекуррентных и свёрточных нейронных сетей

Оболенский Д.М., Шевченко В.И., Ченгарь О.В., Мащенко Е.Н., Соина А.С.
Севастопольский государственный университет,
Россия, 299053, г. Севастополь, ул. Университетская, 33
E-mail: denismaster@outlook.com, VIShevchenko@sevsu.ru, OVChengar@sevsu.ru,
ENMaschenko@sevsu.ru, ASSoina@sevsu.ru

Аннотация. В данной статье авторы рассматривают проблему классификации сообществ в социальной сети ВКонтакте. Исследовано применение нейронных сетей для классификации групп пользователей по степени радикальности. В работе построена модель нейронной сети с рекуррентной долговременной памятью (LSTM) с использованием современных программных технологий и методологий. Результирующая модель обучается на тестовом наборе данных, а также оценивается с использованием выбранных показателей, таких как F1, точность и потери. Модель свёрточной нейронной сети также строится и оценивается. Впоследствии эти модели сравнивают между собой. Модель, основанная на свёрточных нейронных сетях, имеет более высокое значение метрик, чем модель, основанная на модели LSTM. Предлагаются методы предварительной обработки данных, а также использование фреймворка Keras для Python с бэкэндом Tensorflow для построения классификаторов нейронных сетей. Полученная в результате свёрточная сетевая модель применима к основному набору данных при поиске радикальных сообществ в социальных сетях. Проведены исследования модели и представлен анализ результатов.

Ключевые слова: социальные сети, свёрточные нейронные сети, рекуррентные нейронные сети, Python, Keras.

Для цитирования: Оболенский Д.М., Шевченко В.И., Ченгарь О.В., Мащенко Е.Н., Соина А.С. 2021. Реализация классификатора групп в социальных сетях с помощью рекуррентных и свёрточных нейронных сетей. Экономика. Информатика, 48 (1): 100–115. DOI: 10.52575/2687-0932-2021-48-1-100-115.

An implementation of social network group classification model based on recurrent and convolution neural networks

Denis M. Obolensky, Victoria I. Shevchenko, Olga V. Chengar,
Elena N. Maschenko, Anastasia S. Soina
Sevastopol State University, 33 Universitetskaya St, Sevastopol, 299053, Russia
E-mail: denismaster@outlook.com, VIShevchenko@sevsu.ru, OVChengar@sevsu.ru,
ENMaschenko@sevsu.ru, ASSoina@sevsu.ru

Abstract. In this article, the authors consider the problem of classifying communities in the social network VKontakte. The application of neural networks for the classification of user groups according to the degree of radicality is investigated. In the work, a model of a recurrent long short term memory (LSTM) neural network is built using modern software technologies and methodologies. The resulting model is trained on a test dataset and is also evaluated using the selected metrics, such as F1, accuracy and loss. A convolutional neural network model is also built and evaluated. Subsequently, these models are compared with each other. The model which is based on convolutional neural networks had the higher value of metrics, than one based on the LSTM model. Methods for preprocessing data, as well as using the Keras framework for Python with

Tensorflow backend for building neural network classifiers are proposed. The resulting convolutional network model is applicable to the core dataset when searching for radical communities on social networks. The research of the model is carried out and the analysis of the results is presented.

Keywords: social networks, convolutional neural networks, recurrent neural networks, Python, Keras.

For citation: Obolensky D.M., Shevchenko V.I., Chengar O.V., Maschenko E.N., Soina A.S. 2021. An implementation of social network group classification model based on recurrent and convolution neural networks. Economics. Information technologies, 48 (1): 100–115 (in Russian). DOI: 10.52575/2687-0932-2021-48-1-100-115.

Введение

Поиск и анализ сообществ пользователей является важным инструментом изучения и анализа социальных сетей, позволяющим исследовать модульную организацию сети и использовать полученную информацию для решения различных задач [Мащенко и др., 2020]. К примеру, знания о структуре сообществ необходимы для предсказания связей и атрибутов пользователей, расчёта близости пользователей в социальном графе, оптимизации потоков данных в социальной сети, внедрения механизмов влияния на сообщества и т. д. [Мошкин, Андреев, 2019].

Автоматический анализ и классификация сообществ и групп в социальной сети позволят выявлять радикальные группы пользователей, источники конфликтов в социальных сетях, а также применять мягкие меры погашения конфликтов. В частности, в работе Мащенко и др. [Мащенко и др., 2020] показано применение поиска «радикального», «нейтрального» и «объединяющего» контента, в том числе при помощи автоматического анализа контента сообществ. Подобные данные могут быть собраны в автоматическом режиме при помощи портала «Открытые данные» [Оболенский и др., 2020]

Современные способы классификации и анализа тональностей текста применяются в основном к сообщениям пользователей [Котельников, Клековкина, 2012], в то время как не учитывается информация о самом сообществе и его тематике.

Ранее был предложен метод классификации сообществ на основе их описания и контента с помощью простых методов машинного обучения [Оболенский и др., 2020], однако данный способ использует лишь наиболее простые методы [Коусари, Хейдирисафа, 2019].

В рамках данной статьи предлагается практический способ построения классификатора групп и сообществ в социальных сетях на основе нейросетевых методов машинного обучения [Коусари, Хейдирисафа, 2019], в частности, с использованием рекуррентной сети Long-Short Term Memory (LSTM) и свёрточной сети (Convolutional Neural Network, CNN) [Смирнова, Шишков, 2016], реализуемый при помощи современных программных фреймворков, в частности Keras и Tensorflow.

Разработка и обучение классификатора на основе рекуррентной нейронной сети

LSTM был разработан для преодоления проблем простой рекуррентной сети (RNN), позволяя сети хранить данные в своего рода памяти, к которой она может получить доступ позже. LSTM – это особый тип рекуррентной нейронной сети (RNN), который может изучать долгосрочные шаблоны [Рамасундарам, 2010]

Структура сети представлена на рис. 1.

Ключевым понятием в сети LSTM является ячейка состояния. Данное состояние обновляется дважды, что в результате стабилизирует градиенты. Также существует скрытое состояние, которое действует как короткосрочная память. Это показано на рис. 2.

В структуре LSTM можно выделить Forget Gate, Input Gate, Output Gate [Zhang, 2015]. Рассмотрим их подробнее. На рис. 3 показана структура Forget Gate.

На первом этапе необходимо определить, какая информация должна быть сброшена из ячейки состояния. Для этого используется скрытый слой с сигмовидной функцией активации, который называется *Forget Gate* [Zhang, 2015].

На следующем шаге необходимо определить, какую новую информацию мы хотим сохранить в ячейке состояния. Он состоит из двух частей [Zhang, 2015].

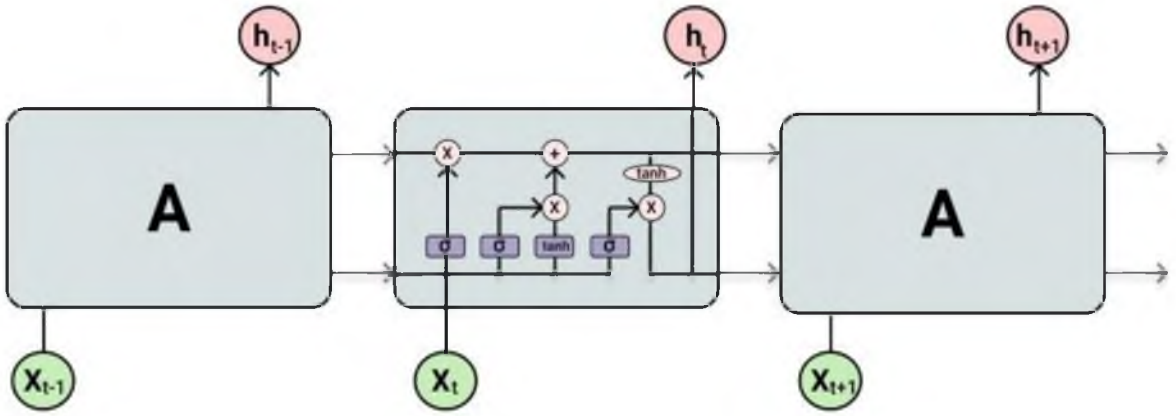

Рис. 1. Структура нейронной сети LSTM
Fig. 1. LSTM structure

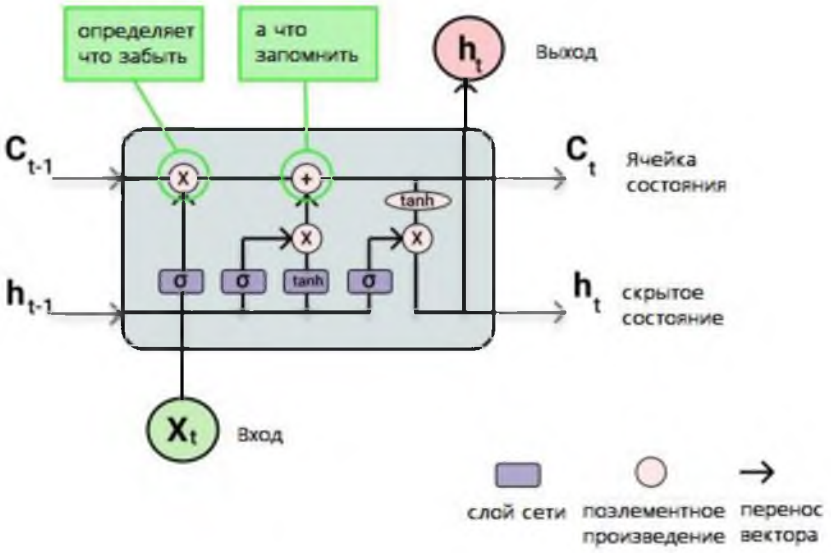

Рис. 2. Структура ячейки сети LSTM
Fig. 2. LSTM Cell structure

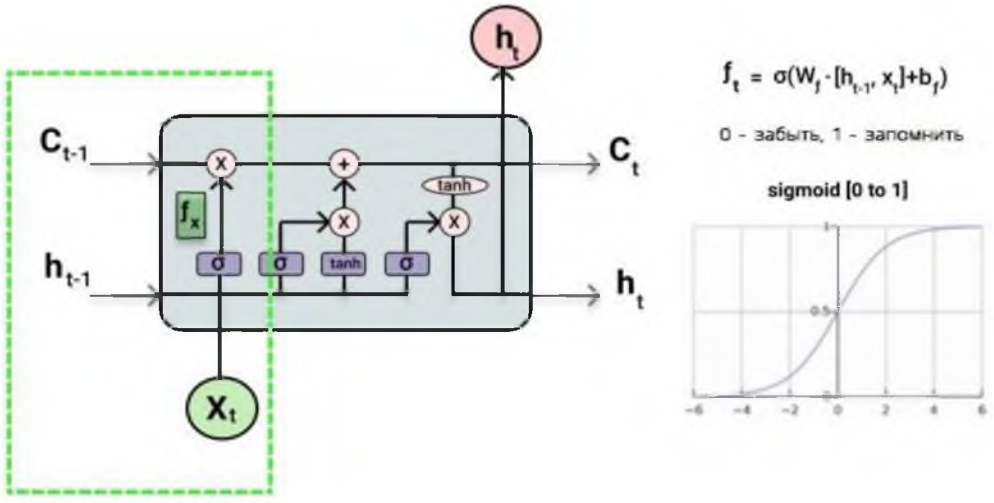

Рис. 3. Forget Gate
Fig. 3. Forget Gate

Первая часть – сигмовидный слой, называемый также *Input Gate*, он определяет, какие значения будут обновляться (рис. 4). Далее – *tanh*-слой, необходимый для расчета потенциальных значений, которые могут быть добавлены в состояние. На следующем этапе мы обновляем ячейку состояния новым значением (рис. 5).

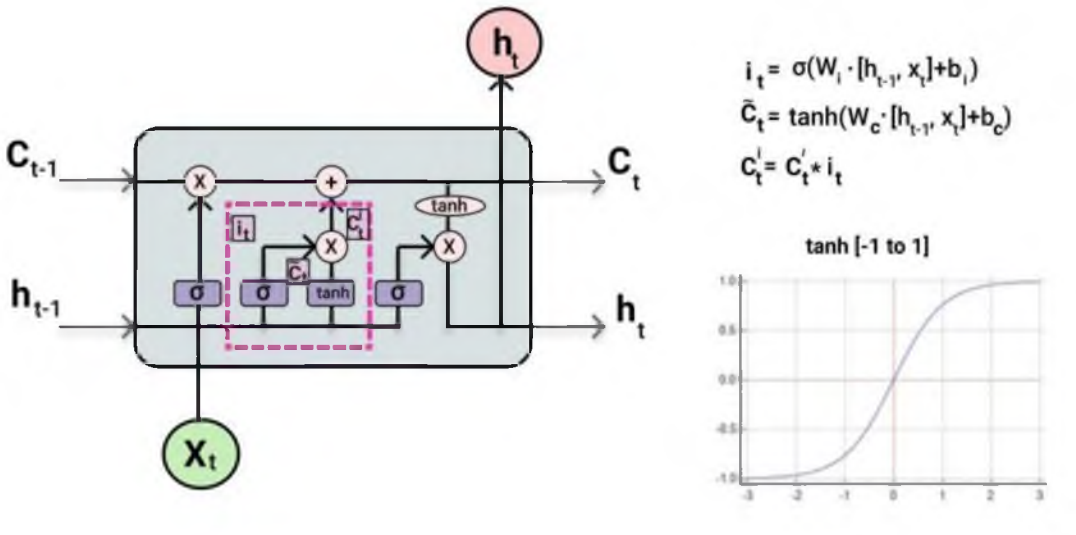

Рис. 4. Input Gate
Fig. 4. Input Gate

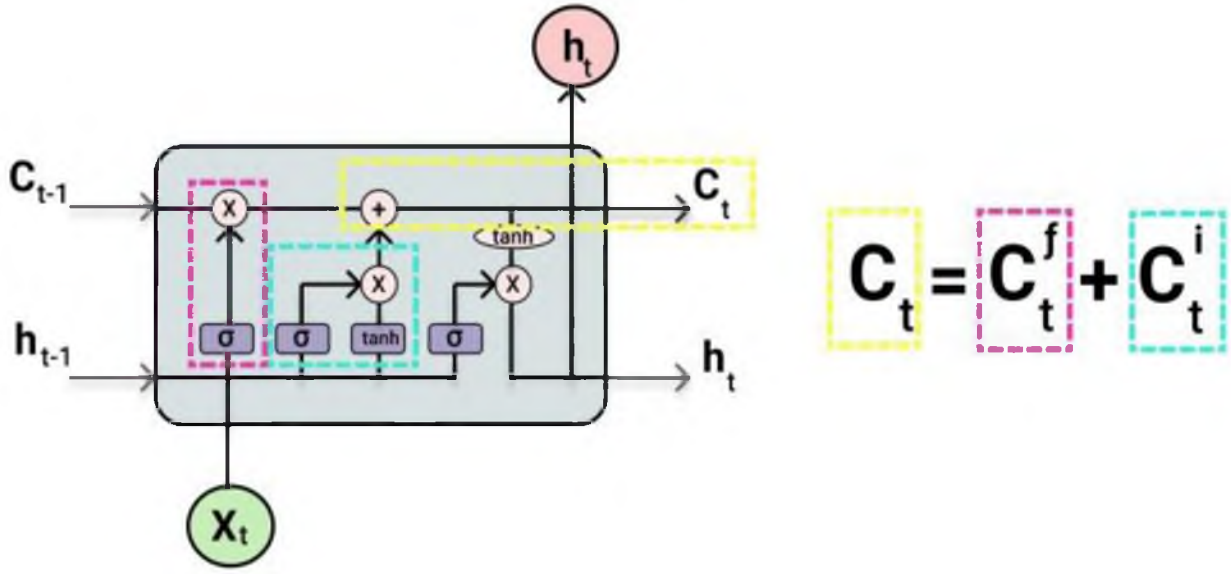

Рис. 5. Процесс обновления ячейки состояния
Fig. 5. Cell State update process

Далее определяется, какое значение будет выходным. Это выходное, результирующее значение определяется на основе ячейки состояния, но при этом проходит через ряд фильтров на основе сигмовидного и тангенциального слоев (рис. 6) [Zhang, 2015].

Подобные нейросети могут активно запоминать скрытые зависимости во входных данных и часто используются в задачах генерации и классификации текстовой информации [Джо, 2010].

Для реализации нейросети на практике использовался фреймворк Keras. Keras — высокоуровневый фреймворк, работающий поверх TensorFlow, позволяющий решать большое количество рутинных задач текстовой классификации, например, преобразование текста в числовые последовательности для анализа и быстрое моделирование нейронных сетей для машинного обучения [Батура, 2017].

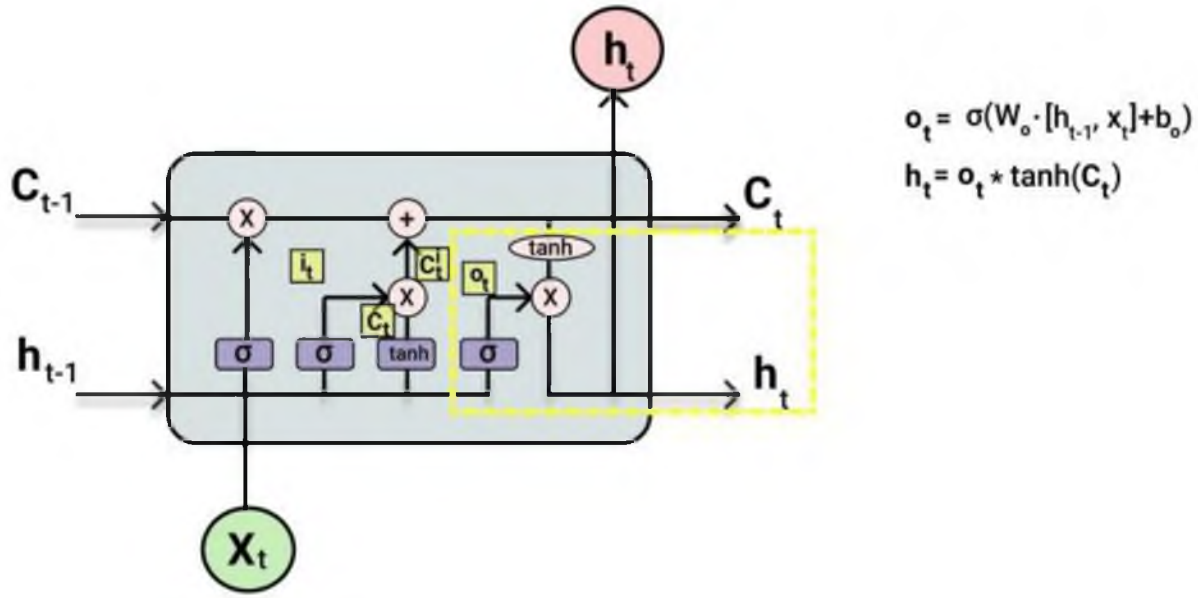

Рис. 6. Output Gate
Fig. 6. Output Gate

Для применения алгоритма текстовой классификации необходимо произвести предварительную обработку набора данных – как обучающего, так и целевого.

Так как нейросеть не может работать с текстовыми данными напрямую, то нужно составить словарь всех слов, заменить каждое слово на число — уникальный номер слова в словаре, а также выровнять длину каждого текста до нужного размера, обычно для этого берется максимальная длина одного элемента входных данных. Далее на представленных данных применяется любой алгоритм классификации [Шевелев, 2006].

Для решения поставленной выше задачи преобразования текста в числовые вектора, использовался специальный класс *Tokenizer* из фреймворка Keras. Применяя полученный словарь на предварительно обработанных данных, были сформированы числовые вектора.

Полученные числовые вектора – разной размерности, поэтому также требуется определить максимальную длину фразы [Багутдинов и др., 2020].

Полученный датасет разбивается при помощи функции *train_test_split()* из библиотеки *sklearn* на обучающее и тестовое подмножество. При этом данная функция производит перемешивание данных, таким образом, как в обучающем, так и в тестовом наборах присутствуют данные всех классов.

Так как векторы чисел – разной размерности, необходимо привести их к одному виду. Для этого используют технику *padding* – выбирается нужная длина вектора, все вектора меньшей длины – дополняются нулевыми значениями, все вектора большей длины – обрезаются. Взяв длину наиболее длинной фразы за основу (для предотвращения потери информации из-за обрезания длины вектора), можно применить данную технику на полученных числовых векторах [Рубцова, 2015; Нгуен, Шираи, 2013].

В свою очередь, метки классов (ранее представленные как числа) необходимо преобразовать в двоичные вектора, причем длина такого вектора – количество классов, на всех позициях стоят 0, на позиции соответствующего класса – единицы. Для преобразования нужно воспользоваться специальной функцией *to_categorically()*.

Так как данные несбалансированные, то в качестве метрики не используется метрика *accuracy* – процент верного распознавания. Допустимой метрикой в задаче классификации текстов с учётом несбалансированного набора данных является метрика *F1* [13]. В библиотеке Keras метрика *F1* по умолчанию не представлена, поэтому ее надо реализовать вручную. Класс, реализующий пользовательскую *F1*-метрику, показан на рис. 7.

```
In [26]: from keras import backend as K

def f1(y_true, y_pred):
    def recall(y_true, y_pred):
        """Recall metric.

        Only computes a batch-wise average of recall.

        Computes the recall, a metric for multi-label classification of
        how many relevant items are selected.
        """
        true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
        possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
        recall = true_positives / (possible_positives + K.epsilon())
        return recall

    def precision(y_true, y_pred):
        """Precision metric.

        Only computes a batch-wise average of precision.

        Computes the precision, a metric for multi-label classification of
        how many selected items are relevant.
        """
        true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
        predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
        precision = true_positives / (predicted_positives + K.epsilon())
        return precision
    precision = precision(y_true, y_pred)
    recall = recall(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall+K.epsilon()))
```

Рис. 7. Пользовательская реализация метрики F1 для фреймворка Keras
Fig. 7. Implementing the custom F1 metric

Используя фреймворк Keras, можно построить нейронную сеть. Для реализации сети в Keras существует ряд API (*Sequential* и *Functional*). Использование *Sequential* позволяет реализовать сеть гораздо более простым образом, при этом *Functional* API намного более гибкое и мощное. Данные API взаимозаменяемые и могут дополнять друг друга. Код для построения модели показан на рис. 8.

```
model = Sequential()
model.add(Embedding(max_features + 1, maxSequenceLength))
model.add(LSTM(32, dropout=0.3))
model.add(Dense(num_classes, activation='sigmoid'))
```

Рис. 8. Определение модели LSTM-сети с помощью Sequential API
Fig. 8. LSTM network code created with Keras Sequential API

На первом этапе создается новая модель путем создания объекта класса *Sequential*. Далее в модель добавляется новый *Embedding*-слой, а за ним – LSTM-слой с коэффициентом *Dropout* = 0.3. Далее – скрытый слой с сигмовидной функцией активации.

Далее модель необходимо скомпилировать с выбранной функцией потерь, оптимизатором и метриками. Нужно указать также реализованную ранее метрику F1.

Также в составе модели был использован оптимизационный алгоритм *adam*, функция потерь на основе категориальной кросс-энтропии, а также выбраны метрики accuracy и разработанная выше метрика F1. Подобная структура модели наиболее полно отвечает задачам текстовой классификации при помощи нейронных сетей [Абрамов; Эспиндола и др.]. Итоговая архитектура и количество параметров сети представлены на рис. 9.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 484)         24779832
_________________________________________________________________
lstm (LSTM)                  (None, 32)                66176
_________________________________________________________________
dense (Dense)                (None, 3)                 99
=================================================================
Total params: 24,846,107
Trainable params: 24,846,107
Non-trainable params: 0
_________________________________________________________________
None
```

Рис. 9. Архитектура скомпилированной Keras-модели
Fig. 9. Keras compiled model architecture

Очередной этап = обучение полученной модели на обучающем множестве. Для этого был выбран размер *batch* равным 32 [Котельников, 2012]. Обучение производилось в течении 3-х эпох. Также использовалось 10 % обучающего множества как валидационное множество. Для обучения модели используется метод *fit*.

Графики изменения метрик accuracy, f1, а также функции потерь в зависимости от эпохи представлены ниже на рис. 10, рис. 11 и рис. 12 соответственно. На вертикальной оси показано значение соответствующей метрики, на горизонтальной – эпоха обучения. Значение каждой метрики представлено в интервале [0,1]. Чем больше значение метрики и меньше значение функции потерь – тем точнее получается итоговая модель.

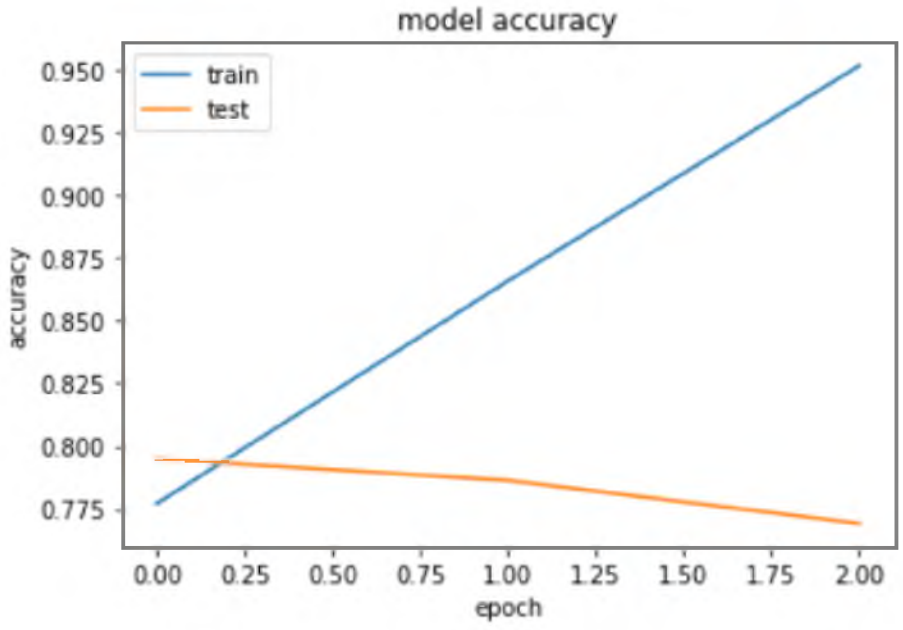

Рис. 10. Изменение метрики Accuracy в зависимости от эпохи
Fig. 10. LSTM Accuracy over time

Видно, что при обучении сети происходит повышение точности модели, а значение функции потерь убывает. При валидации модели на тестовом множестве точность немного снижается в зависимости от эпохи обучения, а значение функции потерь немного повышается.

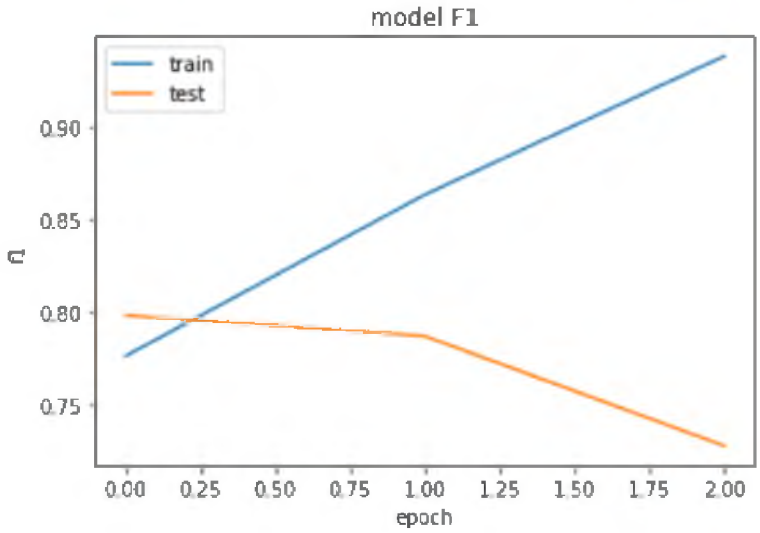

Рис. 11. Изменение метрики F1 в зависимости от эпохи
Fig. 11. LSTM F1 metric value over time

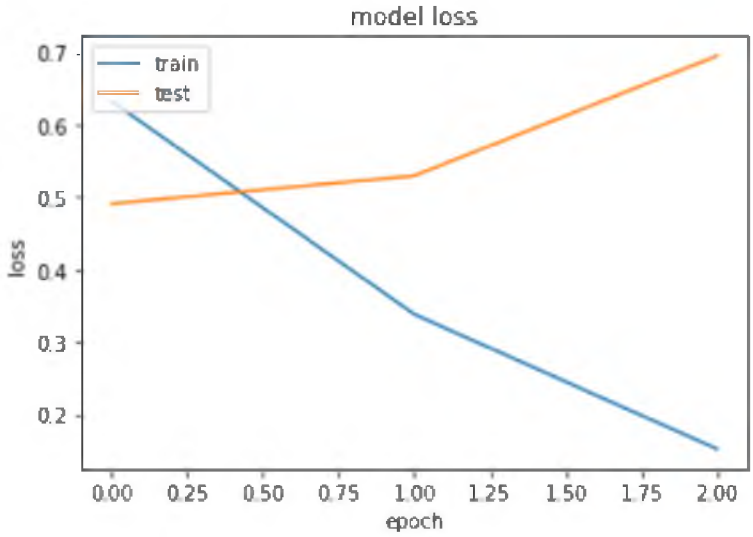

Рис. 12. Изменение значения функции потерь в зависимости от эпохи
Fig. 12. LSTM Loss over time

Разработка и обучение классификатора на основе свёрточной нейронной сети

Свёрточные нейронные сети (CNN) успешно зарекомендовали себя в задачах распознавания и классификации изображений, а также используются в других для повышения точности результатов в сравнении с традиционными методами [Кузьмицкий, 2013]. Свёрточная нейронная сеть – это особый тип нейронных сетей прямого распространения. Обычная данная сеть состоит из 1 или нескольких свёрточных слоев, а также 1 или нескольких скрытых слоев. Количество слоев варьируется в зависимости от задачи [Коллури, Разия, Наяк, 2020].

Основная особенность таких сетей – наличие чередующихся слоев типа «свёртка – субдискретизация», таких слоев может быть несколько. Операция свёртки происходит следующим образом: каждый фрагмент входа поэлементно умножается на матрицу весов (ядро), а результат – суммируется. Эта сумма является элементом выхода, он также называется картой признаков. Взвешенная сумма входов пропускается через функцию активации (рис. 13).

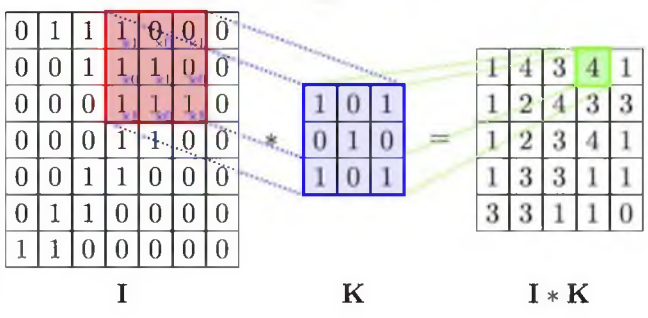

Рис. 13. Работа свёрточного слоя CNN-сети
Fig. 13. Convolution layer of the CNN network

Модель свёрточной сети также можно реализовать с помощью Keras [Жанг, 2015]. Для этого также используется Sequential модель. Первым слоем выступает Embedding-слой [Maac и др., 2011]. Далее следует слой Dropout. После него – свёрточный слой (Conv1D) с оконной функцией 5x5 и функцией активации ReLU [Сянг Жанг, 2015].

Также используется слой субдискретизации GlobalMaxPooling1D. После него используется скрытый полносвязный слой и выходной слой с функцией активации *softmax*.

Программная реализация данной модели представлена на рис.14 ниже:

```
model = Sequential()
model.add(Embedding(max_features + 1, maxSequenceLength))
model.add(Dropout(0.2))
model.add(Conv1D(128, 5,
                 padding='valid',
                 activation='relu',
                 strides=1))
model.add(GlobalMaxPooling1D())
model.add(Dense(128))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
```

Рис. 14. Реализация CNN-сети с помощью Keras
Fig. 14. CNN network code created with Keras Sequential API

Полученная модель компилируется с помощью метода *compile*. Используются аналогичные метрики, функция оптимизации и функция потерь, как и в случае с LSTM:

Архитектура реализованной свёрточной сети представлена на рисунке 15. Можно заметить, что число параметров, используемых в сети, больше, чем в сети LSTM, но незначительно.

Для обучения сети также применяется метод *fit*, но используются 4 эпохи обучения.

Метод evaluate применён для проверки качества модели.

Графики изменения метрик accuracy, f1, а также функции потерь в зависимости от эпохи представлены на рис. 16, рис. 17 и рис. 18 соответственно.

Полученную модель свёрточной нейронной сети можно сохранить при помощи метода *save*() модели. Также необходимо сохранить и объект *tokenizer*, так как он используется для обработки слов.

Видно, что динамика изменения значений функций метрик и функции потерь похожа на аналогичные динамики в сети LSTM, однако можно сделать вывод, что итоговые значения функции потерь ниже, чем у LSTM, а значения метрик выше.

Сравнив результаты F1-метрик разных моделей (простая, LSTM и CNN), можно составить следующую сравнительную таблицу (табл. 1.).

Таким образом, наиболее эффективным и точным классификатором является классификатор на основе свёрточной нейронной сети (CNN).

```
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=[f1, 'acc'])

print (model.summary())
```

```
Собираем модель...
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, None, 484)         24779832
_________________________________________________________________
dropout (Dropout)            (None, None, 484)         0
_________________________________________________________________
conv1d (Conv1D)              (None, None, 128)         309888
_________________________________________________________________
global_max_pooling1d (Global (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               16512
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_5 (Dense)              (None, 3)                 387
=================================================================
Total params: 25,106,619
Trainable params: 25,106,619
Non-trainable params: 0
_________________________________________________________________
None
```

Рис. 15. Компиляция Keras-модели
Fig. 15. CNN model compilation

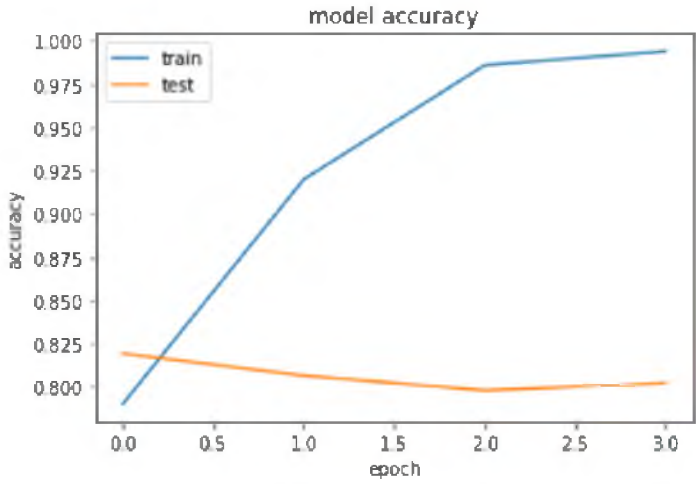

Рис. 16. Изменение метрики Accuracy в зависимости от эпохи
Fig. 16. CNN model accuracy over time

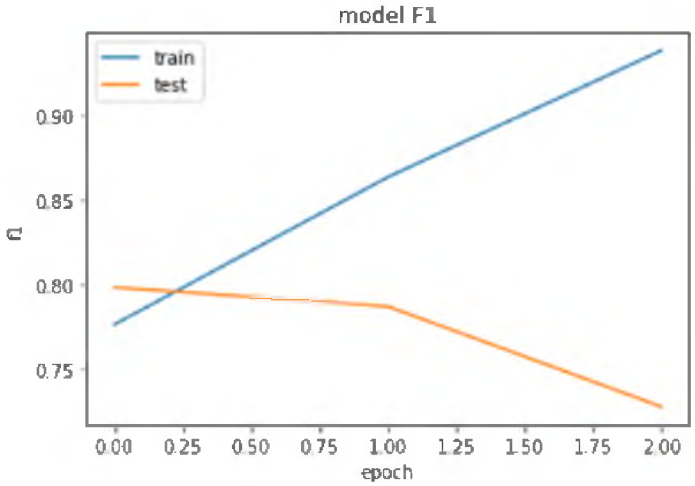

Рис. 17. Изменение метрики F1 в зависимости от эпохи
Fig. 17. CNN model F1 metric value over time

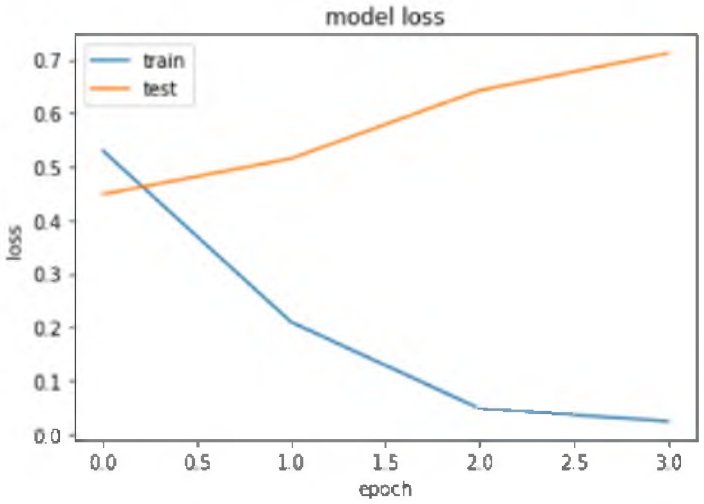

Рис. 18. Изменение значения функции потерь в зависимости от эпохи
Fig. 18. CNN model loss over time

Таблица 1
Table 1

Сравнение метрик F1 различных классификаторов
Classifiers comparison by F1 metric value

Название классификатора	Значение метрики F1
На основе алгоритма машинного обучения SGD	0.5294
На основе рекуррентной нейронной сети LSTM	0.7335
На основе свёрточной нейронной сети CNN	0.7954

Заключение

В результате применения классификатора на основе свёрточной нейронной сети весь основной набор данных был размечен. Был добавлен новый столбец label со значением равным соответствующей предсказанной метке класса.

Была произведена аналитика полученных в ходе классификации данных, результаты классификации представлены в табл. 2.

Таблица 2
Table 2

Результаты применения классификатора
Classification results

Расшифровка показателя	Обозначение	Значение показателя
1	2	3
Всего записей для разбора	N	9635185
С признаком "event"		791825
С признаком "page"		3311246
Группы до 10 участников		1341261
Группы от 10 до 100 участников		3506878
Группы от 100 до 1000 участников		3291257
Группы без описания, пришлось просмотреть "вручную"		3177212
Не участвуют в рассмотрении (признак 0)		38480
Всего не участвуют в рассмотрении (признак 0,4,5)	Nпуст	8769333
Всего участвуют в рассмотрении	Nобщ	865852
Число нейтральных записей	Nнейтр	779412
Число радикальных записей	Nрад	6181

Окончание табл. 2

1	2	3
Число примеряющих записей	Nприм	80259
	Nобщ/N	0,09
	Nнейтр/Nобщ	0,900167696
	Nрад/Nобщ	0,007138633
	Nприм/Nобщ	0,092693671

Соотношение рассматриваемых и не подходящих для исследования данных показано на рис. 19. Можно сделать вывод, что всего 9 % данных подходят для анализа. 91 % данных являются непригодными для исследований и не используются.

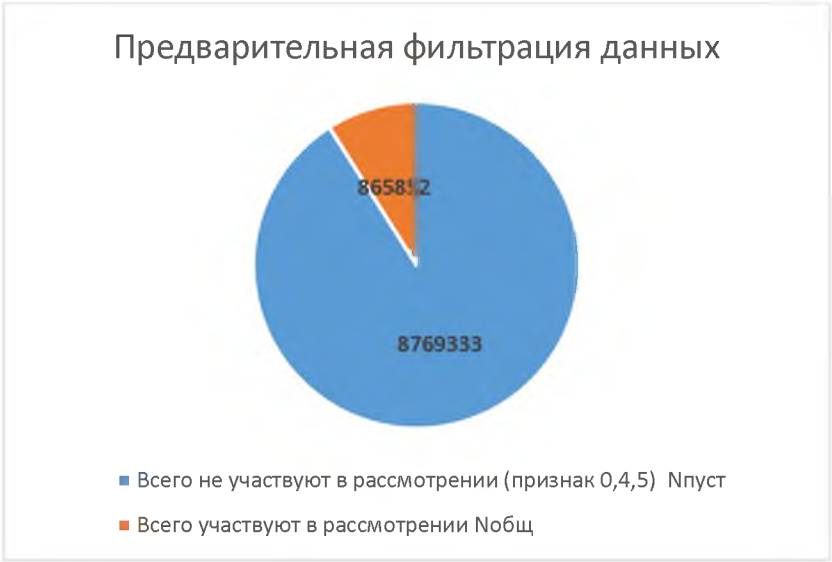

Рис. 19. Соотношение всех записей
Fig. 19. The ratio of useful data and not suitable data for research

Анализируя соотношение полезных записей, был сделан вывод, что всего 0,7 % групп в среднем классифицированы, как «радикальные», примерно 9 % – как «примиряющие», а остальные 90,1 % – как «нейтральные».

Таким образом, в рамках данной статьи успешно применен алгоритм классификации текстовых данных с помощью нейронных сетей, в частности, рекуррентной и свёрточной, в рамках задачи обработки текстовых данных социальных групп на основе их описания и других параметров для задачи текстовой небинарной классификации.

Для нормализации входных данных была проведена предварительная обработка данных, в частности, изменение регистра текста, удаление спецсимволов и стэмминг.

Были построены модели классификатора на основе рекуррентной и свёрточной нейронных сетей с помощью библиотек TensorFlow и Keras, а также определены метрики данных моделей. Установлено, что свёрточная нейронная сеть выдает лучшие результаты. Для оценки данных моделей и их сравнения использовалась метрика F1.

Выполнена аналитика на основном наборе данных с целью поиска радикальных, нейтральных и объединяющих сообществ.

В дальнейших работах планируется исследование взаимосвязи пользователей в предсказанных с помощью разработанных моделей радикальных сообществах, взаимосвязи пользователей между собой. Также будет произведено построение графа пользователей и его анализ с помощью Gephi.

Рис. 20. Соотношение полезных записей
Fig. 20. Ratio of neutral, positive and radical data

Список литературы

1. Абрамов Р.А. Основные метрики задач классификации в машинном обучении. URL: https://webiomed.ai/blog/osnovnye-metriki-zadach-klassifikatsii-v-mashinnom-obuchenii/ (дата обращения: 12.08.2020).

2. Багутдинов Р.А., Саргсян Н.А., Красноплахтыч М.А. 2020. Аналитика, инструменты и интеллектуальный анализ больших разнородных и разномасштабных данных. Экономика. Информатика. 47 (4): 792–802. DOI 10.18413/2687-0932-2020-47-4- 792-802.

3. Батура Т.В. 2017. Методы автоматической классификации текстов. Программные продукты и системы. 1 (30): 85–99. DOI: 10.15827/0236-235X.030.1.085-099.

4. Котельников Е.В. 2012. Комбинированный метод автоматического определения тональности текста. Программные продукты и системы. 3 (109): 189–195.

5. Котельников Е.В., Клековкина М.В. 2011. Автоматический анализ тональности текстов на основе методов машинного обучения. РОМИП. URL: http://www.dialog-21.ru/media/1380/105.pdf. (дата обращения: 12.08.2020).

6. Кузьмицкий Н.Н. 2013. Создание универсальных классификаторов текстовых образов на основе сверточных нейросетевых технологий. ГрафиКон'2013: 23-я Международная конференция по компьютерной графике и зрению: 234–238.

7. Мащенко Е.Н, Оболенский Д.М., Соина А.С., Ченгарь О.В. 2020. «Разобщающий», «нейтральный» и «объединяющий» контент сообществ сети ВКонтакте на Украине: определение понятий. The Newman in Foreign Policy. 54 (98): 38–43.

8. Мошкин В.С., Андреев И.А. 2019. Сравнение эффективности применения алгоритмов сентимент-анализа неструктурированных ресурсов социальных сетей. Восьмая Междунар. конф. «Системный анализ и информационные технологии». САИТ-2019: тр. конф. М.: ФИЦ ИУ РАН: 534–540.

9. Метрики в задачах машинного обучения. URL: https://habr.com/ru/company/ods/blog/328372/ (дата обращения: 12.08.2020).

10. Оболенский Д.М. и др. 2020. Сбор и предварительная обработка данных пользователей социальных сетей с помощью портала «Открытые Данные». Modern Science. 7 (2): 369–378.

11. Рубцова Ю.В. 2015. Построение корпуса текстов для настройки тонового классификатора. Программные продукты и системы. 1 (109): 72–78. DOI:10.15827/0236-235X.109.072-078.

12. Смирнова О.С., Шишков В.В. 2016. Выбор топологии нейронных сетей и их применение для классификации коротких текстов. International Journal of Open Information Technologies. 4 (8): 50–54.

13. Шевелев О.Г., Петраков А.В. 2006. Классификация текстов с помощью деревьев решений и нейронных сетей прямого распространения. Вестник Томского государственного университета. 290 (1): 300–307.

14. Espíndola, Rogério & Ebecken, Nelson. 2005. On extending f-measure and g-mean metrics to multi-class problems. Sixth international conference on data mining, text mining and their business

applications. 35: 25-34. URL: https://www.researchgate.net/publication/286914533_On_extending_f-measure_and_g-mean_metrics_to_multi-class_problems (дата обращения: 12 августа 2020).

15. Jo T. 2010. NTC (Neural Text Categorizer): Neural Network for Text Categorization. International Journal of Information Studies. 2(2): 83–96.

16. Kolluri J., Razia S., Nayak S.R. 2020. Text Classification Using Machine Learning and Deep Learning Models. URL: https://ssrn.com/abstract=3618895 (дата обращения: 12 августа 2020).

17. Kowsari J. M., Heidarysafa M. 2019. Text Classification Algorithms: A Survey. Information, 10 (4): 150–154.

18. Maas A.L., Daly R.E., Pham P.T., Huang D., Ng A.Y., Potts C. 2011. Learning word vectors for sentiment analysis. The International Language Technologies. 1: 142–150.

19. Nguyen T.H., Shirai K. 2013. Text Classification of Technical Papers Based on Text Segmentation. Natural Language Processing and Information Systems. NLDB 2013. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 121.

20. Ramasundaram S., Victor. S. 2010. Text Categorization by Backpropagation Network. International Journal of Computer Applications. 8 (6): 1–5.

21. Zhang, X. 2015. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. In Advances in Neural Information Processing Systems. 2015. Feb. 649–657 p.

22. Understanding LSTM Networks. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs (дата обращения: 17 July 2020).

23. Nguyen T.H., Shirai K. 2013. Text Classification of Technical Papers Based on Text Segmentation. In: Métais E., Meziane F., Saraee M., Sugumaran V., Vadera S. (eds) Natural Language Processing and Information Systems. NLDB 2013. Lecture Notes in Computer Science, vol. 7934. Springer, Berlin, Heidelberg.

24. Ramasundaram, S., Victor, S. Text Categorization by Backpropagation Network. International Journal of Computer Applications. 2010. 8 (6): 1–5.

25. Zhang, X. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. In Advances in Neural Information Processing Systems. 2015. Feb. 649–657 p.

26. Understanding LSTM Networks. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs (accessed: 17 July 2020).

References

1. Abramov R. Osnovnye metriki zadach klassifikacii v mashinnom obuchenii. URL: https://webiomed.ai/blog/osnovnye-metriki-zadach-klassifikatsii-v-mashinnom-obuchenii/ (accessed: 12 August 2020) (in Russian).

2. Bagutdinov R.A., Sargsan N.A., Krasnoplakhtych M.A. 2020. Analytics, tools and intellectual analysis of large different and differential data. Economics. Information technologies. 47 (4): 792–802 (in Russian). DOI 10.18413/2687-0932-2020-47-4-792-802.

3. Batura T.V. 2017. Metody avtomaticheskoj klassifikacii tekstov. Programmnye produkty i sistemy. 1 (30): 85–99 (in Russian). DOI: 10.15827/0236-235X.030.1.085-099.

4. Kotel'nikov E.V. 2012. Kombinirovannyj metod avtomaticheskogo opredelenija tonal'nosti teksta. Programmnye produkty i sistemy. 3 (109): 189–195 (in Russian).

5. Kotel'nikov E.V., Klekovkina M.V. 2011. Avtomaticheskij analiz tonal'nosti tekstov na osnove metodov mashinnogo obuchenija. ROMIP. URL: http://www.dialog-21.ru/media/1380/105.pdf. (accessed: 12 August 2020) (in Russian).

6. Kuz'mickij, N.N. 2013. Sozdanie universal'nyh klassifikatorov tekstovyh obrazov na osnove svertochnyh nejrosetevyh tehnologij. GrafiKon'2013: 23-ja Mezhdunarodnaja konferencija po komp'juternoj grafike i zreniju: 234–238 (in Russian).

7. Mashhenko E.N, Obolenskij D.M., Soina A.S., Chengar' O.V. 2020. "Razobshhajushhij", "nejtral'nyj" i "obedinjajushhij" kontent soobshhestv seti VKontakte na Ukraine: opredelenie ponjatij. The Newman in Foreign Policy. 54 (98): 38–43 (in Russian).

8. Moshkin V.S., Andreev I.A. 2019. Sravnenie jeffektivnosti primenenija algoritmov sentiment-analiza nestrukturirovannyh resursov social'nyh setej. Vos'maja Mezhdunar. konf. "Sistemnyj analiz i informacionnye tehnologii". САИТ-2019:тр. конф. М.: ФИЦ ИУ РАН: 534–540 (in Russian).

9. Metriki v zadachah mashinnogo obuchenija. URL: https://habr.com/ru/company/ods/blog/328372/ (accessed: 12 August 2020) (in Russian).

10. Obolensky D.M. et el. 2020. Sbor i predvaritel'naja obrabotka dannyh pol'zovatelej social'nyh setej s pomoshh'ju portala "Otkrytye Dannye". Modern Science. 7 (2): 369–378 (in Russian).

11. Rubcova Ju.V. 2015. Postroenie korpusa tekstov dlja nastrojki tonovogo klassifikatora. Programmnye produkty i sistemy. 1 (109): 72–78 (in Russian). DOI:10.15827/0236-235X.109.072-078

12. Smirnova O. S., Shishkov V. V. 2016. Vybor topologii nejronnyh setej i ih primenenie dlja klassifikacii korotkih tekstov. International Journal of Open Information Technologies. 4 (8): 50–54 (in Russian).

13. Shevelev O.G., Petrakov A.V. 2006. Klassifikacija tekstov s pomoshh'ju derev'ev reshenij i nejronnyh setej prjamogo rasprostranenija. Vestnik Tomskogo gosudarstvennogo universiteta. 290 (1): 300–307 (in Russian).

14. Espíndola, Rogério & Ebecken, Nelson. 2005. On extending f-measure and g-mean metrics to multi-class problems. Sixth international conference on data mining, text mining and their business applications. 35: 25–34. URL: https://www.researchgate.net/publication/286914533_On_extending_f-measure_and_g-mean_metrics_to_multi-class_problems (access: 12 August 2020).

15. Jo T. 2010. NTC (Neural Text Categorizer): Neural Network for Text Categorization. International Journal of Information Studies. 2 (2): 83–96.

16. Kolluri J., Razia S., Nayak S.R. 2020. Text Classification Using Machine Learning and Deep Learning Models. URL: https://ssrn.com/abstract=3618895 (дата обращения: 12 августа 2020).

17. Kowsari J. M., Heidarysafa M. 2019. Text Classification Algorithms: A Survey. Information, 10 (4): 150–154.

18. Maas A.L., Daly R.E., Pham P.T., Huang D., Ng A.Y., Potts C. 2011. Learning word vectors for sentiment analysis. The International Language Technologies. 1: 142–150.

19. Nguyen T.H., Shirai K. 2013. Text Classification of Technical Papers Based on Text Segmentation. Natural Language Processing and Information Systems. NLDB 2013. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 121.

20. Ramasundaram S., Victor. S. 2010. Text Categorization by Backpropagation Network. International Journal of Computer Applications. 8 (6): 1–5.

21. Zhang, X. 2015. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. In Advances in Neural Information Processing Systems. 2015. Feb. 649–657 p.

22. Understanding LSTM Networks. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs (accessed: 17 July 2020).

23. Nguyen T.H., Shirai K. 2013. Text Classification of Technical Papers Based on Text Segmentation. In: Métais E., Meziane F., Saraee M., Sugumaran V., Vadera S. (eds) Natural Language Processing and Information Systems. NLDB 2013. Lecture Notes in Computer Science, vol. 7934. Springer, Berlin, Heidelberg.

24. Ramasundaram, S., Victor, S. Text Categorization by Backpropagation Network. International Journal of Computer Applications. 2010. 8 (6): 1–5.

25. Zhang, X. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. In Advances in Neural Information Processing Systems. 2015. Feb. 649–657 p.

26. Understanding LSTM Networks. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs (accessed: 17 July 2020).

ИНФОРМАЦИЯ ОБ АВТОРАХ

Оболенский Денис Михайлович, ассистент кафедры информационных технологий и компьютерных систем ФГАОУ ВО «Севастопольский государственный университет», г. Севастополь, Россия

Шевченко Виктория Игоревна, кандидат технических наук, доцент, зав. базовой кафедрой корпоративных информационных систем ФГАОУ ВО «Севастопольский государственный университет», г. Севастополь, Россия

INFORMATION ABOUT THE AUTHORS

Denis M. Obolensky, Assistant of the Department of Information Technologies and Computer Systems, Sevastopol State University, Sevastopol, Russia

Victoria I. Shevchenko, Candidate of technical sciences, Associate Professor, Head of the Base Department of Corporate Information Systems Sevastopol State University, Sevastopol, Russia

Ченгарь Ольга Васильевна, кандидат технических наук, доцент, доцент кафедры информационных технологий и компьютерных систем ФГАОУ ВО «Севастопольский государственный университет», г. Севастополь, Россия

Olga V. Chengar, Candidate of technical sciences, Associate Professor, Assitance Professor of the Department of Information Technologies and Computer Systems, Sevastopol State University, Sevastopol, Russia

Мащенко Елена Николаевна, кандидат технических наук, доцент, доцент кафедры информационных технологий и компьютерных систем ФГАОУ ВО «Севастопольский государственный университет» г. Севастополь, Россия

Elena N. Maschenko, Candidate of technical sciences, Associate Professor, Assitance Professor of the Department of Information Technologies and Computer Systems, Sevastopol State University Sevastopol, Russia

Соина Анастасия Сергеевна, кандидат технических наук, ассистент ФГАОУ ВО «Севастопольский государственный университет», г. Севастополь, Россия

Anastasia S. Soina, Candidate of technical sciences, Sevastopol State University, Sevastopol, Russia