УДК 004.032.26

А. А. Сайгин [1], Н. П. Плотникова [1]

[1] Мордовский государственный университет имени Н. П. Огарёва, г. Саранск, Российская Федерация

ВЕКТОРИЗАЦИЯ НОРМАТИВНО – СПРАВОЧНОЙ ИНФОРМАЦИИ С ПОМОЩЬЮ МОДЕЛИ НЕЙРОННОЙ СЕТИ BERT

Аннотация. В статье описывается векторизация нормативно–справочной информации с помощью Bidirectional Encoder Representations from Transformers (BERT) – нейронной сети для обработки естественного языка. Рассматривается архитектура нейронной сети Transformer, ее принцип работы. Описывается архитектура нейронной сети BERT, ее использование с библиотекой Transformers. Приводится пример программного кода для использования модели на практике. Проводится оценка работы нескольких моделей на базе описанной архитектуры, поддерживающих русский язык, методом определения похожести слов. Описывается составление датасета для оценки работы моделей. Сравниваются результаты оценки работы разных моделей.

Ключевые слова: BERT, transformers, обработка естественного языка, векторизация, метод определения похожести, корреляция.

A. A. Saygin [1], N. P. Plotnikova [1]

[1] Ogarev Mordovia State University Saransk, Russian Federation

VECTORIZATION OF REGULATORY - REFERENCE INFORMATION USING THE BERT NEURAL NETWORK

Abstract. The article describes a vectorization normative and reference information using Bidirectional Encoder Representations from Transformers (BERT) – a neural network for natural language processing. The architecture of the transformer neural network and the principle of its operation are considered. The architecture of the neural network BERT is described, its use with the Transformers library. An example of a program code for using the model in practice is given. The work of several models based on the described architecture supporting the Russian language is assessed by the method of determining the similarity of words. The compilation of a dataset for evaluating the performance of models is described. The results of evaluating the performance of different models are compared.

Key words: BERT, transformers, natural language processing, vectorization, method of determining similarity, correlation.

BERT (Bidirectional Encoder Representations from Transformers) — это нейронная сеть для обработки естественного языка, разработанная Google и использующая архитектуру Transformer, основанную на механизме внимания. Суть механизма внимания состоит в том, что при чтении текста невольно выделяются ключевые слова, которые несут наибольшую семантическую нагруженность [1]. С помощью BERT можно создавать программы для обработки естественного языка: чат-боты, автоматические переводчики, классификаторы и кластеризаторы текстов.

Трансформер (Transformer) – это нейронная сеть, которая для повышения скорости обучения использует механизм внимания. Внутри она состоит из двух компонентов – кодирующего и декодирующего. Механизм внимания позволяет декодирующему компоненту сфокусироваться на конкретном значении, прежде чем сгенерировать результат. Поскольку данный механизм усиливает сигнал от релевантной части входной последовательности, то результат работы таких моделей лучше по сравнению с остальными. Кодирующий компонент представляет собой набор энкодеров (кодирующих элементов), которые располагаются последовательно друг за другом, формируя собой стек. Декодирующий компонент аналогичен по структуре и представляет из себя стек декодирующих элементов – декодеров. Количество декодеров равно количеству энкодеров [2].

BERT – это обученный стек энкодеров Трансформера. У моделей BERT'а есть большое количество слоев энкодера. Все энкодеры идентичны по структуре и имеют два подслоя.

Входная последовательность, поступающая в энкодер, сначала проходит через слой внутреннего внимания (self-attention). Данный слой позволяет посмотреть на другие слова во входящем предложении во время кодирования конкретного слова. Выход слоя внутреннего внимания отправляется в нейронную сеть прямого распространения (feed-forward neural network). Для каждого слова в предложении она применяется независимо [1].

На базе BERT было обучено большое количество моделей, различающихся количеством параметров и внутренних слоев. Они разрабатывались с целью либо улучшить точность, либо ускорить вычисления. XLNet от Google/CMU и RoBERTa от Facebook обладают улучшенной точностью, а DistilBERT от HuggingFace быстрее работает.

Для упрощения работы как с оригинальной моделью BERT, так и с ее модификациями, группа исследователей из HuggingFace разработала библиотеку Transformers. Она способна работать со всеми архитектурами для задач обработки естественного языка (BERT, GPT-2, RoBERTa, XLM, DistilBert, XLNet), а также предоставляет возможность работы с PyTorch (оригинальная модель была разработана с использованием TensorFlow) [3]. Есть возможность конвертации модели из одного фреймворка в другой. Вместе с библиотекой появилась платформа, на которой разработчики могут делиться своими моделями. Всего в библиотеке более 32 предобученных моделей для более 100 языков.

Векторизация текста осуществляется при помощи библиотеки Transformers. На ее базе можно реализовать класс, который будет принимать преобразуемый текст и возвращать готовые векторы. При этом не будет требоваться предварительная обработка текста, поскольку все будет выполняться внутри класса.

Класс содержит два поля – tokenizer и model (рис. 1.). tokenizer – это объект токенизатора текста. Он загружается из предобученной модели с помощью метода from_pretrained, определенного в классе AutoTokenizer. model – это объект модели, выполняющий преобразование токенов в вектора. Она загружается из предобученной модели с помощью метода from_pretrained, определенного в классе AutoModel. На самом деле для каждой из архитектур определены свои классы токенизаторов и моделей (например, BertTokenizer и BertModel для BERT или XLNetTokenizer и XLNetModel для XLNet). Но может возникнуть ситуация, когда архитектура модели изначально неизвестна. Поэтому в библиотеке и определены классы AutoTokenizer и AutoModel, которые определяют архитектуру используемой модели и вызывают подходящие конструкторы [4].

```
def __init__(self, model_name_or_path):
    self.tokenizer = transformers.AutoTokenizer.from_pretrained(model_name_or_path)
    self.model = transformers.AutoModel.from_pretrained(model_name_or_path)
```

Рис. 1. Инициализация модели

Первым этапом текст разбивается на токены с помощью объекта tokenizer. К исходной строке добавляются две специальные метки. [SEP] помечает конец предложения. [CLS] ставится в начало строки, это специальный токен, используемый в задачах классификации текста [1]. Токены представляют собой три массива – ‘input_ids’ с индексами токенов в словаре модели, 'token_type_ids' с токенами-разделителями (поскольку в нашей задаче будет всего одна метка [SEP], то массив будет заполнен нулями и в нашей задаче он не используется), и 'attention_mask' с токенами, указывающие на значимые слова (рис. 2.).

Далее полученные токены подаются на вход модели, где они преобразуются в трехмерную матрицу с числовыми значениями токенов. Первое измерение матрицы соответствует входным предложениям, второе – токенам, на которые предложения разбивались, третье – выходу модели (рис. 3.). На вход модели поступают torch-тензоры, поэтому необходимо преобразовать массивы с токенами (рис. 4.).

```python
def text_tokenization(self, dataframe):
    result = []
    max_len = 0
    for title in dataframe:
        title = " ".join(title.split())
        inputs = self.tokenizer.encode_plus(
            title,
            None,
            add_special_tokens=True,
            return_token_type_ids=True
        )
        result.append(inputs.data)
        if len(inputs.data['input_ids']) > max_len:
            max_len = len(inputs.data['input_ids'])
    for i in result:
        i['input_ids'] = i['input_ids'] + [0] * (max_len - len(i['input_ids']))
        i['token_type_ids'] = i['token_type_ids'] + [0] * (max_len - len(i['token_type_ids']))
        i['attention_mask'] = i['attention_mask'] + [0] * (max_len - len(i['attention_mask']))
    return result
```

Рис. 2. Разбиение текста на токены

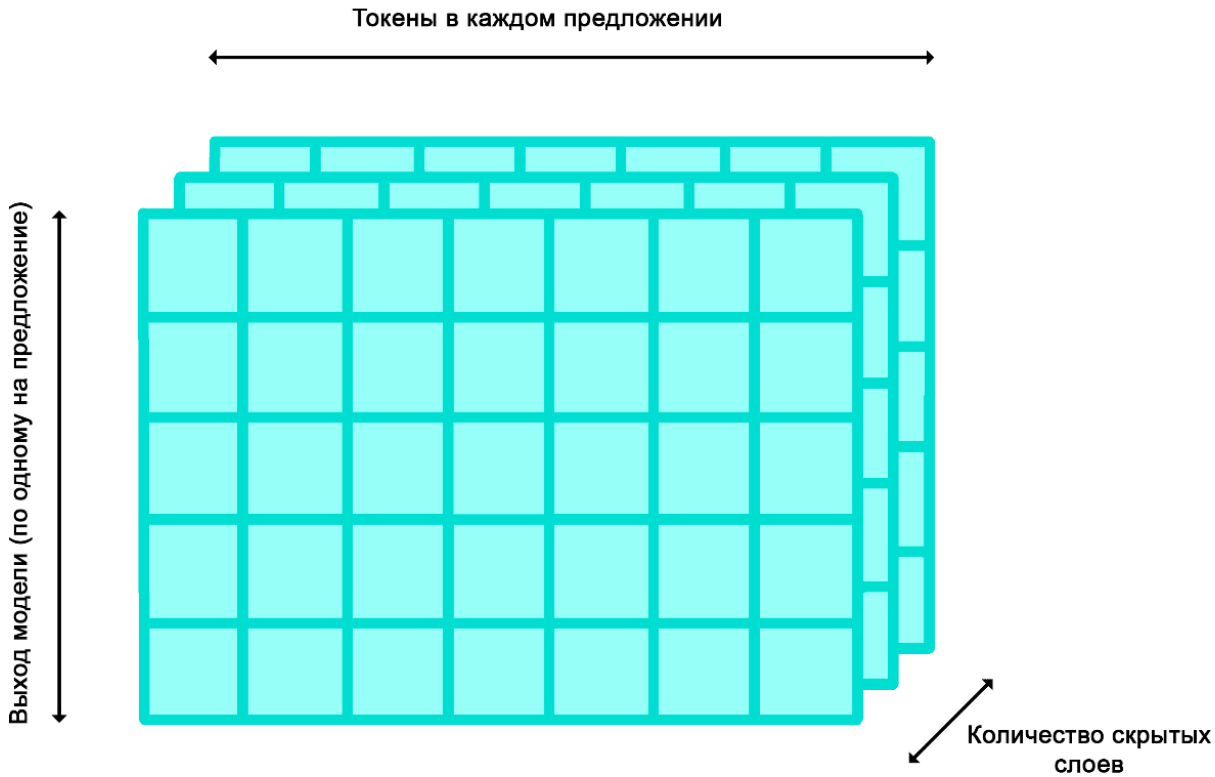

Рис. 3. Выход модели BERT’a

```
55  def text_vectorization(self, dataframe):
        tokenized = self.text_tokenization(dataframe)

        input_ids = []
        attention_mask = []
        for i in tokenized:
            input_ids.append(i['input_ids'])
            attention_mask.append(i['attention_mask'])
        input_ids = torch.tensor(numpy.asarray(input_ids), dtype=torch.long)
        attention_mask = torch.tensor(numpy.asarray(attention_mask), dtype=torch.long)

        with torch.no_grad():
            last_hidden_states = self.model(input_ids, attention_mask=attention_mask)

        result = last_hidden_states[0].numpy()

        return result
```

Рис. 4. Преобразование токенов в векторы

Для классификации предложений необходим только выход BERT'a для токена [CLS]. В матрице значения, соответствующие данному токену, находятся на первой позиции каждого предложения (рис. 5.). Эти значения нужно отделить от остальной матрицы (рис. 6.).

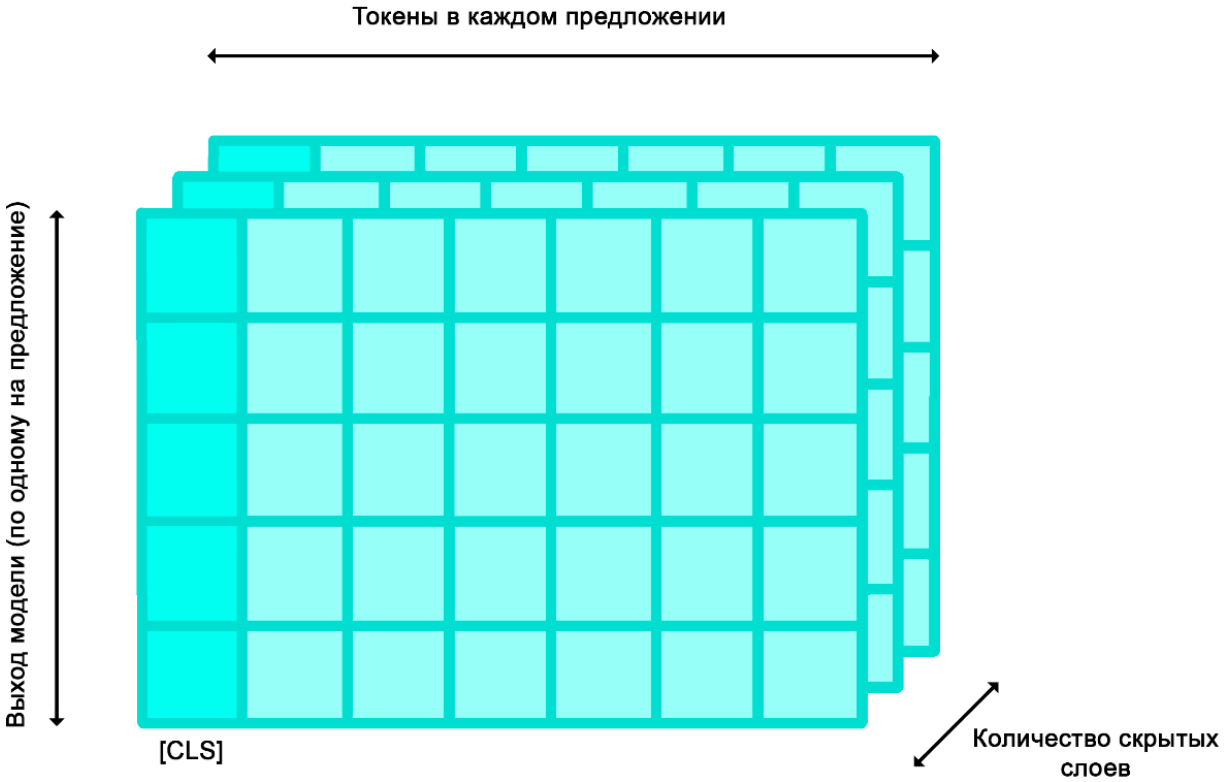

Рис. 5. Выделение значений токена [CLS]

```
def text_vectorization_cls(self, dataframe):
    vectorized = self.text_vectorization(dataframe)
    result = vectorized[:, 0, :]
    return result
```

Рис. 6. Преобразование токенов в векторы и выделение значений токена [CLS]

Для оценки качества работы модели применим метод определения похожести слов [5]. Суть данного метода в том, что используется датасет, в котором эксперты определили похожести пар слов, словосочетаний или предложений. У векторов, полученных после обработки входных пар фраз моделью, оценивается их похожесть. Далее вычисляется значение коэффициента корреляции между полученными значениями и эталонными, который будет показателем качества работы модели.

Для оценки похожести векторов чаще всего применяется косинусное расстояние. Дополнительно, для решения данной задачи используются значения Евклидова и Манхэттенского расстояний [6]. Они являются обратными к похожести величинами, поэтому для корректного вычисления коэффициента корреляции используются их отрицательные значения (рис. 7.).

```
def compare_texts_cosine(self, text1, text2):
    vector1 = self.text_vectorization_cls([text1])
    vector2 = self.text_vectorization_cls([text2])
    cosine = numpy.dot(vector1[0], vector2[0]) / (numpy.linalg.norm(vector1[0]) * numpy.linalg.norm(vector2[0]))
    return cosine

def compare_text_euclid(self, text1, text2):
    vector1 = self.text_vectorization_cls([text1])
    vector2 = self.text_vectorization_cls([text2])
    euclid = numpy.linalg.norm(vector1 - vector2) * -1
    return euclid

def compare_text_manhattan(self, text1, text2):
    vector1 = self.text_vectorization_cls([text1])
    vector2 = self.text_vectorization_cls([text2])
    manhattan = numpy.abs(vector1 - vector2).sum() * -1
    return manhattan
```

Рис. 7. Оценка похожести векторов

Проверка качества работы модели осуществляется посредством специального датасета. Для его составления было выбрано 10 случайных наименований товаров из разных Интернет-магазинов. Из них были составлены пары, каждой из которых несколько экспертов дали свою оценку их сходства по шкале от 1 до 10, где 1 – минимальное сходство, а 10 – полное совпадение. Финальная оценка вычислялась как среднее от всех оценок экспертов. В итоге получается датасет из 55 записей, в которых содержатся две строки и оценка их похожести (рис. 8.).

При оценке схожести пары наименований обе строки подаются на вход модели BERT. На выходе получаются два вектора, соответствующие входным строкам. У полученных векторов находится похожесть. Таким образом обрабатываются все пары строк. Показателем качества работы модели является корреляция между значениями оценок схожести экспертов и вычисленной похожестью полученных векторов. Для нахождения данного значения используются коэффициенты корреляции Пирсона и Спирмана [6].

Машина-перевертыш "Танк", работает от батареек, цв МИКС	Машина-перевертыш "Танк", работает от батареек, цв МИКС	10
Машина-перевертыш "Танк", работает от батареек, цв МИКС	Робот "Бади", ездит по линии, световые эффекты, работает от	5
Машина-перевертыш "Танк", работает от батареек, цв МИКС	Набор для купания "Транспорт": мини-коврик + губка + игруш	2
Машина-перевертыш "Танк", работает от батареек, цв МИКС	Часы настенные, серия: Транспорт, "Спортивное авто", 28х28	2
Машина-перевертыш "Танк", работает от батареек, цв МИКС	Компрессор автомобильный Торнадо АС-580, 14 А, 150 PSI, 30	1
Машина-перевертыш "Танк", работает от батареек, цв МИКС	Массажер «машинка» 2 ролика 2632951	1

Рис. 8. Датасет для оценки работы моделей

Для оценки были выбраны следующие модели, поддерживающие русский язык:

- bert-base-multilingual-cased
- distilbert-base-multilingual-cased
- xlm-mlm-xnli15-1024
- xlm-mlm-17-1280
- DeepPavlov/rubert-base-cased

Результаты оценки работы моделей представлены в таблице 1.

Таблица 1.

Оценки работы моделей

Модель	Оценка	Корреляция	
		Пирсона	Спирмана
bert-base-multilingual-cased	cosine	0.7962	0.4491
	Euclid	0.8839	0.4514
	Manhattan	0.8883	0.4582
distilbert-base-multilingual-cased	cosine	0.8692	0.6551
	Euclid	0.9191	0.6603
	Manhattan	0.9179	0.6585
xlm-mlm-xnli15-1024	cosine	0.6351	0.5797
	Euclid	0.7911	0.5942
	Manhattan	0.8478	0.632
xlm-mlm-17-1280	cosine	0.8468	0.6032
	Euclid	0.8922	0.6199
	Manhattan	0.8907	0.6185
DeepPavlov/rubert-base-cased	cosine	0.8411	0.5816
	Euclid	0.9023	0.568
	Manhattan	0.9048	0.5752

Таким же способом проводится дополнительная оценка работы модели. Отличие от предыдущей состоит в том, что из датасета убираются пары одинаковых наименований, у которых экспертная оценка равна 10 (всего их десять).

Результаты дополнительной оценки работы моделей (без пар одинаковых наименований) представлены в таблице 2.

Таблица 2.

Оценки работы моделей (без пар одинаковых наименований)

Модель	Оценка	Корреляция	
		Пирсона	Спирмана
bert-base-multilingual-cased	cosine	0.07145	-0.0046
	Euclid	0.1534	-0.001133
	Manhattan	0.1845	0.01133
distilbert-base-multilingual-cased	cosine	0.4079	0.3738
	Euclid	0.4457	0.38
	Manhattan	0.4362	0.3768

Модель	Оценка	Корреляция	
		Пирсона	Спирмана
xlm-mlm-xnli15-1024	cosine	0.1899	0.2358
	Euclid	0.182	0.2595
	Manhattan	0.2372	0.3284
xlm-mlm-17-1280	cosine	0.2377	0.2784
	Euclid	0.2133	0.3064
	Manhattan	0.2096	0.3038
DeepPavlov/rubert-base-cased	cosine	0.3452	0.2374
	Euclid	0.3502	0.2116
	Manhattan	0.3682	0.2248

Качество работы моделей стало существенно ниже. Связано это с особенностями ранжирования при вычислении коэффициентов корреляции. Ранжирование является первым этапом корреляции. Это выдача ранга значениям в зависимости от их положения в отсортированном списке. Корреляция вычисляется для ранжированных значений. Поскольку для одинаковых наименований значения похожести определяются однозначно (все эксперты выдают таким парам оценку 10, а косинусное расстояние между одинаковыми векторами равно 1), то и их ранги определяются однозначно, что оказывает влияние на итоговый коэффициент.

В [6] упоминается, что оригинальная модель BERT от Google плохо подходит для определения похожести текстов. В последующих моделях данная проблема решается, что видно из таблиц.

В целом, оценка похожести наименований показала достаточно хорошие результаты. Из этого можно сделать вывод, что на базе нейронной сети BERT можно создавать различные модели для решения задач обработки естественного языка. Такие модели будут работать с высокой точностью.

Для дальнейших исследований в этой области стоит изучить тонкую настройку моделей BERT. Она подстроит модель под входные данные, с которыми предстоит работать, и позволит получать более точные результаты в узких сферах применения обработки естественного языка.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Devlin J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding //arXiv preprint arXiv:1810.04805v2 – 2019.

2. Vaswani A. et al. Attention Is All You Need //arXiv preprint arXiv:1706.03762v5 – 2017.

3. Wolf T. et al. Transformers: State-of-the-Art Natural Language Processing //arXiv preprint arXiv:1910.03771v5 – 2019.

4. The Hugging Face Team. Transformers [Электронный ресурс]. – Режим доступа: https://huggingface.co/transformers/index.html, свободный. – (дата обращения: 08.02.2021).

5. Kalyan KS Sangeetha S. SECNLP: A Survey of Embeddings in Clinical Natural Language Processing //arXiv preprint arXiv:1903.01039v4 – 2020.

6. Reimers N. Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks //arXiv preprint arXiv:1908.10084v1 – 2019.

REFERENCES

1. Devlin J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding //arXiv preprint arXiv:1810.04805v2 – 2019.

2. Vaswani A. et al. Attention Is All You Need //arXiv preprint arXiv:1706.03762v5 – 2017.

3. Wolf T. et al. Transformers: State-of-the-Art Natural Language Processing //arXiv preprint arXiv:1910.03771v5 – 2019.

4. The Hugging Face Team. Transformers [Электронный ресурс]. – Режим доступа: https://huggingface.co/transformers/index.html, свободный. – (дата обращения: 08.02.2021).

5. Kalyan KS Sangeetha S. SECNLP: A Survey of Embeddings in Clinical Natural Language Processing //arXiv preprint arXiv:1903.01039v4 – 2020.

6. Reimers N. Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks //arXiv preprint arXiv:1908.10084v1 – 2019.

Информация об авторах

Андрей Александрович Сайгин – студент кафедры «Автоматизированные системы обработки информации и управления», Мордовский государственный университет имени Н. П. Огарева, г. Саранск, e-mail: andrexsai@mail.ru

Наталья Павловна Плотникова – к. т. н., кафедра «Автоматизированные системы обработки информации и управления», Мордовский государственный университет имени Н. П. Огарева, г. Саранск, e-mail: linsierra@yandex.ru

Authors

Andrey Aleksandrovich Saygin - student of the Subdepartment "Automated information processing and control systems", Mordovia State University named after N.P. Ogarev, Saransk, e-mail: andrexsai@mail.ru

Natalya Pavlovna Plotnikova - Ph.D., Subdepartment of Automated Information Processing and Control Systems, Mordovia State University named after N.P. Ogarev, Saransk, e-mail: linsierra@yandex.ru

Для цитирования

Сайгин А.А., Плотникова Н.П. Векторизация нормативно – справочной информации с помощью модели нейронной сети BERT // «Информационные технологии и математическое моделирование в управлении сложными системами»: электрон. науч. журн. – 2021. – №2(10). – С. 52-59 – DOI: 10.26731/2658-3704.2021.2(10).52-59 – Режим доступа: https://ismm.irgups.ru/toma/210-2021, свободный.. – Загл. с экрана. – Яз. рус., англ. (дата обращения: 29.04.2021)

For citations

Saygin A.A., Plotnikova N.P. Vectorization of regulatory - reference information using the BERT neural network // Informacionnye tehnologii i matematicheskoe modelirovanie v upravlenii slozhnymi sistemami: ehlektronnyj nauchnyj zhurnal [Information technology and mathematical modeling in the management of complex systems: electronic scientific journal], 2021. No. 2. P. 52-59. DOI: 10.26731/2658-3704.2021.2(10).52-59 [Accessed 29/04/21]