DOI https://doi.org/10.15407/usim.2019.01.041
УДК 004.04.043; 004.912; 004.62

Е.А. БОБРОВНИК, магистрант, кафедра укр. языка и прикладной лингвистики,
Ин-т филологии Киевского национального университета имен Тараса Шевченко
бульвар Тараса Шевченко, 14, Киев 01601, Украина,
mailkatherine.bobrovnik@gmail.com

К.К. ДУХНОВСКАЯ, ассистент, кафедра прикладных информационных технологий,
Киевский национальный университет имени Тараса Шевченко,
просп. Академика Глушкова, 4, Киев 03022, Украина,
duchnov@ukr.net

Н.В. ПИРОГ, ассистент, кафедра прикладных информационных технологий,
Киевский национальный университет имени Тараса Шевченко,
просп. Академика Глушкова, 4, Киев 03022, Украина,
mykola.pyroh@ukr.net

ТЕМАТИЧЕСКАЯ КЛАССИФИКАЦИЯ УКРАИНОЯЗЫЧНЫХ ТЕКСТОВ, ТРУДНОСТИ ЕЕ ВНЕДРЕНИЯ

Построены классификаторы украиноязычных текстов методами Random Forest Classifier, Support Vector Machines, Naive Bayes Classifier.u Logistic Regression. Для тренировки этих классификаторов использовался метод контролированного обучения. Суть этого метода заключается в том, что для обучения используется уже готовый классифицированный набор текстов, в качестве которого выступают тексты Брауновский корпуса украинского языка. Лучшие результаты показала модель для классификации украиноязычных текстов на основе метода опорных векторов. Ее средняя точность — 0,80.

Ключевые слова: *классификатор текстовых документов, корпус документов, метод контролированного обучения.*

Введение

Одной из важных задач искусственного интеллекта, информационного поиска и систем текстовой обработки есть задача классификации.

В статье под классификацией понимаем сортировку текстовых документов по заданным тематикам.

При разработке классификаторов используют решения информационного поиска и машинного обучения. Главная трудность при автоматической классификации текстовой информации в том, что документ представлен на естественном языке и не относится к структурированным данным.

Решений, представляющих классификацию англоязычных или русскоязычных текстов, много. Все они представляют алгоритмы для построения классификаторов текстовой информации с более или менее хорошими показателями *полноты* и *точности*.

Но работ, в которых описаны алгоритмы построения классификатора текстов, представленных на украинском языке, и в чем их особенность, авторы не нашли.

Цель статьи — построение тематического классификатора украиноязычных текстов на основе Брауновского корпуса украинского языка.

Постановка задачи

Пусть задан словарь $W = \{w_1, w_2, ..., w_N\}$, где N — это мощность словаря (количество слов, содержащихся в словаре) и коллекция текстовых документов: $D = \{d_1, d_2, ..., d_M\}$, M — количество документов в этой коллекции. Причем каждый i-й документ из коллекции имеет вид: $d_i = (t_1, t_2, ..., t_{M,}, C_k)$, где t_j вычисляется согласно векторной мере, C_k — k-я тематика i-го документа.

Каждая тематика задается описанием: $F_k = \{f_1, f_2, ..., f_{VM}\}$. Задача классификации заключается в том, чтобы найти такое отношение $Rc = d_i X F_k$, которое даст возможность отнесения i-го документа к одной из тематик C_k.

Задача классификации текстовых документов состоит из двух частей. Первая часть сформулирована выше. Вторая — это обучение классификатора. Она заключается в формировании $<C, Rc, d_i>$ на основании априорных данных.

Априорные данные для обучения

Разработка качественного тематического классификатора текстовых документов сопровождается многократным проведением экспериментов для обучения классификатора. В процессе этого необходимо использовать большие наборы текстов. Для решения данной и других лингвистических задач создают разнообразные корпуса текстов.

Корпус текстов представляет собой электронный набор текстовых данных (*dataset*) по различным тематикам или по конкретной, отображающий организацию лексической системы конкретного языка по этим тематикам.

Первый такой корпус текстов был создан в Брауновском университете в 1960 г. Он включал в себя 500 текстов длинной по 2000 символов.

На сегодня известны такие текстовые корпуса как Оксфордский корпус английского языка, Корпус современного американского английского языка (*Corpusof Contemporary American English, COCA*). Британский национальный корпус (B*ritish National Corpus, BNC*), Чешский национальный корпус, Национальный корпус русского языка, Корпус русских учебных текстов и др.

Самый большой англоязычный корпус — это Оксфордский корпус английского языка, он насчитывает 2,5 млрд словосочетаний. Корпус современного американского английского языка — самый крупный корпус англоязычных текстов, который находиться в свободном доступе и насчитывает 445 млн словосочетаний. Британский национальный корпус являет собой эталон англоязычных текстов и насчитывает более 100 млн словосочетаний. Среди языков славянской группы выделяется Чешский национальный корпус, который насчитывает 4 млн словосочетаний.

Национальный корпус русского языка состоит из следующих корпусов:

- основной корпус, который включает разные тексты XVIII–XXI веков;
- синтаксический корпус (глубоко аннотированный), содержит вместе с текстами, для каждого предложения текста описание полной его лексической и морфологической структуры;
- параллельный корпус, в котором возможно найти все переводы конкретного слова на русский, или с русского языка;
- газетный корпус, который состоит из новостной подборки и др.

К сожалению, в украинистике корпусов текстов, созданных для обучения текстовых классификаторов, поисковых машин, систем текстового анализа и решения других лингвистических задач, практически нет. Один из известных — Корпус украинского языка, созданный коллективом лаборатории компьютерной лингвистики Института филологии КНУ им. Т. Шевченко [1].

Этот Корпус оснащен пакетами программ, обеспечивающих автоматический и автоматизированный анализ текстов на морфемном, лексическом и синтаксическом уровнях организации и автоматическое составление разнообразных частотных словарей. Непосредственно сам набор текстов для разработки и обучения лингвистических алгоритмов авторами обнаружен не был.

Подходя к вопросу тематической классификации, отвечающей задаче информационного поиска, мы считаем возможным выделить четыре проекта:

- Брауновский корпус украинского языка [2];
- Большой электронный словарь украинского языка [3];
- Русско-украинский словарь [4];
- *LanguageTool API NLP UK* [5].

Брауновский корпус украинского языка (БрУК) находится в свободном доступе. Это дает возможность беспрепятственной работе с накопленной базой. Корпус насчитывает 730 документов — файлов формата .txt с текстом и описанием информации о нем. Также каждый файл начинается с определенной буквы, обозначающей класс, к которому принадлежит текст. Количество классов БрУК фиксировано и равно девяти.

Категории текстов в БрУК и их процентное наполнение предоставлены в таблице.

Изучая принципы устройства БрУК и его категоризацию, авторы руководствовались трудами [6, 7]. Отметим, что в сравнении с оригинальным проектом Брауновского корпуса его украинская интерпретация претерпела значительные изменения в связи с необходимостью адаптировать категоризацию текстов под язык славянской группы, который в лингвистическом аспекте имеет значительные отличия от групп германских языков.

Построение классификаторов украиноязычных текстов

Исходя из сказанного, считаем возможным утверждать, что БрУК — единственный корпус украинских текстов в открытом доступе для

Таблица. **Категории текстов БрУК**

Идент.	Категория	%	Описание текстов
A	Пресса	25	Репортажи, обзоры, статьи, письма в редакцию; национальные и региональные издания; тематические: политика, спорт, общество; экономика и финансы, короткие новости; культура: театр, литература, музыка, танцы
B	Религиозная литература	3	Книги, периодика, брошюры
C	Профессионально-популярная литература	7	Книги и периодика: домоводство, ремесла, «сад и огород», хобби, ремонт и строительство, конструирование, музыка и танцы, домашние животные, спорт, еда и вино, путешествия, фермерство, рабочие профессии и т.п.
D	Эстетические информативные тексты	7	Информативные тексты, не попадающие в другие категории, в частности, биографии, мемуары, эссе, предисловия, личные письма, художественная, критика, рекламные тексты
E	Административные документы	3	Законы, правительственные акты, отчеты организаций/фондов/компаний, официальные письма
F	Научно-популярная литература	5	
G	Научная литература	10	Книги и периодика; естественные и гуманитарные науки, техника и инженерное дело
H	Учебная литература	15	Учебники, пособия, гуманитарные и естественные науки и т.д.
I	Художественные тексты	25	Романы, повести, рассказы, новеллы, по тематике: детективы, фантастика, приключенческая, любовная, юмористическая и т.д.

внедрения алгоритмов классификации украиноязычных текстов.

Для построения классификаторов украиноязычных текстов авторами были использованы следующие методы и алгоритмы: *Random Forest Classifier, Support Vector Machines, Naive Bayes Classifier. и Logistic Regression*.

Random Forest Classifier — классификатор, использующий статистический метод решения задач классификации, основанный на применении решающих деревьев. Одним из достоинств этого классификатора является то, что он допускает легкое распараллеливание, что дает возможность применения его к обработке больших объемов данных. Также преимуществами этого метода является возможность оценки значимости признаков в модели, высокая скорость обучения и легкая интерпретация полученной модели. Недостатки алгоритма: склонность к переобучению, особенно со многими уровнями шумов, большой размер получаемых моделей классификации.

Метод опорных векторов (*Support Vector Machines*) заключается в построении разделяющей гиперплоскости в пространстве классифицируемых множеств и определении позиции входящего вектора относительно этой плоскости. Этот метод используется для задач классификации текстовых документов, таких как назначение категорий (классов) и выявления спама. Метод опорных векторов также играет важную роль во многих областях распознавания рукописных цифр, таких как, например, службы автоматизации почтовой связи. Алгоритмы данного метода предоставляют достаточно высокую точность классификации [10].

Наивный Байесовский классификатор (*Naive Bayes Classifier*) является одним из эталонных, т.е. классификатором с которым сравнивают качество работы других классификаторов. Его работа основана на теореме Байеса. Результат данного классификатора не может быть улучшен, так как во всех случаях, когда возможен однозначный ответ, он его даст, а в тех случаях, когда ответ неоднозначен, результат количественно характеризует степень этой неоднозначности [11].

Методы логистической регрессии (*Logistic Regression*) основаны на анализе связей независимых переменных. Они являются обобщением метода линейной регрессии с использованием *softmax*-функции и применяются в случае, когда зависимая переменная может принимать только конечное множество значений. Основное отличие и преимущество такого подхода от других моделей и алгоритмов — оценка результата, которую можно было бы рассматривать как значение вероятности для определенного класса. Недостатком этого подхода есть неспособность построения гиперплоскости сложного вида и, как следствие, не очень высокая точность распознавания [12].

Используемый авторами метод контролированного обучения, другими словами обучение с учителем, необходим для тренировки (обучения) всех этих классификаторов. Суть этого метода заключается в том, что для обучения используется уже готовый классифицированный набор текстов, в качестве которого будет выступать БрУК.

Классификаторы были реализованы при помощи высокоуровневого языка программирования *Python* на системе контроля пакетов и дистрибутивов с открытым кодом *Anaconda*3. При этом использовались библиотеки для машинного обучения *scikit-learn* с разработанными методами построения классификаторов и их обучения, библиотеки регулярных выражений и другие вспомогательные инструменты. Блок-схема реализации классификатора представлена на рис. 1.

Процесс классификации текстов структурно может быть подан следующими этапами:

- подготовка текстов и создание набора для тренировки классификаторов;
- предварительная обработка текстов:
  - удаление лишних/малозначительных символов и слов, которые могут некорректно влиять на процесс обучения;
  - лематизация слов (приведение их к изначальной форме);
- трансформация текста в векторное представление;
- обучение моделей;

▪ оценка качества точности моделей при помощи перекрестной валидации.

Идентификаторы БрУК использовались авторами в качестве меток для контролируемого обучения, ведь цель классификации состоит в том, чтобы спрогнозировать метку класса (*class label*), что является выбором из предопределенного списка возможных вариантов.

Работа с файлами: открытие текстовых документов и удаление лишней метаинформации, которая была добавлена к текстам БрУК, реализуется авторами при помощи регулярных выражений и дополнительных библиотек (*re* — регулярные выражения, *pynlple* — обработка текстов, *pandas* — для манипулирования данными и их анализа) (рис. 2).

Особенность работы с текстовыми данными, в отличие от численных данных состоит в том, что перед этапом непосредственного машинного обучения необходимо выполнить предварительную обработку, *чистку* текстов от мало информативных слов, символов, знаков пунктуации, ссылок, *html*-тегов и т.п. Это популярная практика, так как после нее точность работы моделей в большинстве случаев только возрастает, а размерность векторов, с которой работает эта модель, уменьшается, что влечет за собой уменьшение трудозатрат. Другой, не менее важный метод обработки текстов, лемматизация терминов — приведение термина к изначальной форме на основании грамматических и морфологических правил естественного языка. Преимущества лемматизации — меньшие объемы текстов, генерация и рост показателей точности классификатора. Недостаток лемматизации — необходимость в словарях языка, на котором предоставляются текстовые документы, таким образом, происходит поиск нормальных форм для всех слов текста, что может существенно снизить скорость работы модели.

Для лемматизации текстов БрУК использовалась библиотека *pymorphy*2, которая создана для обработки русского языка, но содержит в себе также украинские словари для морфологического анализа. Также библиотека *pymorphy*2 предоставляет возможность удалить стоп-слова из текста документа. Стоп-слова — это слова, которые не несут смысловую нагрузку текста и встречаются в каждом текстовом документе (предлоги, союзы и т.д.). Список стоп-слов дополняется списком собственных стоп-слов (рис.3).

Рис. 1. Блок-схема построения классификатора

Для удаления лишних тегов, ссылок, номеров телефонов, электронных адресов и подобного использовалась библиотека *pynlple* (рис.4).

Для формирования вектора признаков текста, проводится частотный анализ, при котором считается количество каждого слова в тексте. В *Classifier* и *Logistic Regression* схема частотного анализа имеет следующий вид:

Document Preprocessing

```
path = 'corpus/data/good'
```

```
text_list = []
tag_list = []
for doc in list(FilePathSource([path], extension_suffix='.txt')):
    doc = doc.replace('\\', '/')
    filename = file_name(doc)
    with open(doc, 'r', encoding='utf8') as text:
        new_text = ''
        for line in text:
            if not re.match('<[^>]*>', line):
                new_text += line
        text_list.append(new_text)
        tag_list.append(filename[0])
```

```
df = pd.DataFrame({

    'text': text_list,
     'tag': tag_list

})
```

Рис. 2. Листинг кода работы с текстовыми файлами

Text Preprocessing

```
import numpy
from sklearn.model_selection import train_test_split
from stop_words import get_stop_words
import pymorphy2
```

```
def lemmatized(text):
    new_txt  = []
    morph = pymorphy2.MorphAnalyzer(lang='uk')
    for i in text.split():
        p = morph.parse(i)[0]
        new_txt.append(p.normal_form)
    return ' '.join(new_txt)
```

```
STOPWORDS = get_stop_words('uk')
exclude = set(string.punctuation)
exclude.update(['..'])

def text_prepare(text):
    text = ' '.join(ch for ch in text.split() if ch not in exclude)
    text = ' '.join(word for word in text.split() if word not in STOPWORDS)
    text = lemmatized(text)
    return text
```

```
STOPWORDS.extend(['а', 'й', 'із', 'бо'])
```

Рис. 3. Листинг кода для «чистки» текста

- построить словарь W из всех слов, которые используются в изначальных текстах D;
- для каждого текста $d_i \in D$ и для каждого термина из словаря $t_j \in W$ подсчитать число вхождений x_{ij} слова t_j в текст D_i.

Таким образом, для каждого текста $t_i \in W$ мы получим вектор целых неотрицательных чисел x_i, длина которого равна количеству слов в словаре.

Это базовый вид метода частотного анализа текстов. Для текстов разного объема величина значений частоты x может сильно отличаться, т.е. чем объемней текст, тем больше может быть повторов слов.

Для уменьшения этого эффекта применяется метод нормализированного частотного анализа или TF (*term frequency*) [8], значение частоты x делится на общее число слов в тексте d_i:

$$TF(d_i, W) = \frac{x(d_i, W)}{size(d_i)}.$$

Данные алгоритмы для своей работы требуют наличие словаря. Формирование словаря — важная составляющая обработки

```
STOPWORDS.extend(['а', 'й', 'із', 'бо'])
```

```
from pynlple.processing.preprocessor import *

stack = [
                                        HtmlEntitiesUnescaper(),
                                        BoldTagReplacer(),
                                        URLReplacer(' urlTag '),
                                        HtmlTagReplacer(' '),
                                        EmailReplacer(' emailTag '),
                                        PhoneNumberReplacer(' phoneNumberTag '),
                                        AtReferenceReplacer(' atRefTag '),
                                        CommaReplacer(),
                                        QuotesReplacer(),
                                        DoubleQuotesReplacer(),
                                        SoftHyphenReplacer(),
                                        DashAndMinusReplacer(),
                                        TripledotReplacer(),
                                        ToLowerer(),
                                        DigitsReplacer(True, '0'),
                                        MultiPunctuationReplacer(),
                                        MultiNewLineReplacer(),
                                        MultiWhitespaceReplacer(),
                                        WordTokenizer(),
                                        Trimmer(),
        ]
```

```
%time
t = StackingPreprocessor(stack)
df['preproc_text'] = df["text"].apply(lambda x: text_prepare(t.preprocess(x))).values
```

```
CPU times: user 6 µs, sys: 0 ns, total: 6 µs
Wall time: 11.2 µs
```

Рис. 4. Листинг кода «чистка» от *html*-тегов

```
Training
```

```
X = df['preproc_text']
y = df['tag']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.05, random_state = 242)
```

Рис. 5. Использование библиотеки *sklearn*

текстов. От того, какие слова там окажутся, зависит информативность векторов-признаков и, соответственно, качество работы системы. Для классификаторов *Naive Bayes Classifier* и *Logistic Regression* словарь формируется из всего набора текстов. Например, предлоги и союзы встречаются в любом тексте. Для решения задачи классификации текстов эти термины не информативны, поэтому их можно удалить из словаря. В то же время, для определения авторства данного текстового документа частота использования предлогов может оказаться весьма полезным свойством.

Возникает вопрос: удалять часто используемые слова или нет? Существует компромиссный вариант формирования вектор-признаков — *TF-IDF* [8]. Данный метод не убирает часто используемые слова из словаря, но уменьшает их вес в вектор-признаке. Для всех слов словаря рассчитывается коэффициент обратной частоты *IDF* (*inverse document frequency*):

$$IDF(t_i) = \log \frac{size(D)}{size(D(t_i))},$$

т.е. логарифм дроби, числитель которой — общее количество документов, а знаменатель — количество документов, которые содержат слово t_i. Следовательно, чем чаще встречается слово, тем меньше его значение *IDF*.

Итоговая частотная характеристика имеет вид произведения частотной характеристики *TF* на коэффициент обратной частоты *IDF*:

$$TF\text{-}IDF(d_i, D, W) = TF(d_i, t_j) * IDF(t_j).$$

Перед приведением к векторному представлению текстов, наборы текстов делятся на учебную и тестовую выборки при помощи библиотеки *sklearn* (рис. 5):

TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
def tfidf_features(X_train, X_test):
    tfidf_vectorizer = TfidfVectorizer(max_df=0.90, min_df=5, ngram_range=(1,2), token_pattern='(\S+)')
    tfidf_vectorizer.fit(X_train)
    X_train = tfidf_vectorizer.transform(X_train)
    X_test  = tfidf_vectorizer.transform(X_test)
    return X_train, X_test, tfidf_vectorizer.vocabulary_
```

```
X_train_tfidf, X_test_tfidf, tfidf_vocab = tfidf_features(X_train, X_test)
```

Рис. 6. Векторное представление тренировочной и тестовой выборки

```
models = [
    RandomForestClassifier(n_estimators=200, max_depth=3, random_state=0),
    LinearSVC(),
    MultinomialNB(),
    LogisticRegression(random_state=0),
]
CV = 5
cv_df = pd.DataFrame(index=range(CV * len(models)))
entries = []
for model in models:
    model_name = model.__class__.__name__
    accuracies = cross_val_score(model, X_train_tfidf, y_train, scoring='accuracy', cv=CV)
    for fold_idx, accuracy in enumerate(accuracies):
        entries.append((model_name, fold_idx, accuracy))
cv_df = pd.DataFrame(entries, columns=['model_name', 'fold_idx', 'accuracy'])
import seaborn as sns
sns.boxplot(x='model_name', y='accuracy', data=cv_df)
sns.stripplot(x='model_name', y='accuracy', data=cv_df,
              size=8, jitter=True, edgecolor="gray", linewidth=2)
plt.show()
```

Рис. 7. Построение моделей классификаторов

Листинг для создания векторного представления тренировочной и тестовой выборки представлен на рис. 6:

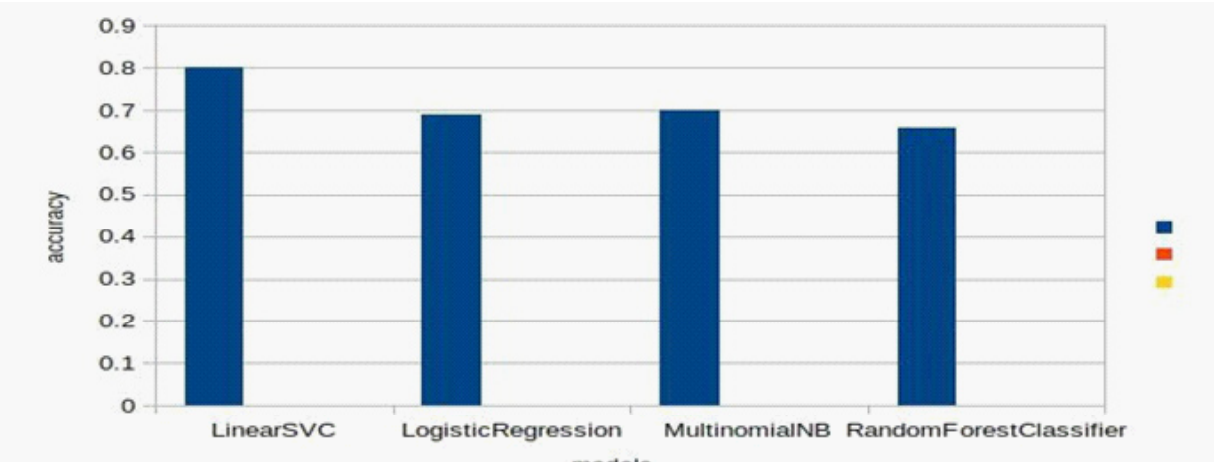
В результате описанной программной реализации были получены четыре модели классификаторов, которые следует обучить, так как эти методы не дают прямого решения задачи классификации. Решение задачи классификации получают после обучения этих моделей на множествах сходных решений, в процессе чего выявляются эмпирические закономерности в текстовых документах.

Обучение и тестирование моделей классификаторов

Обучение классификаторов — этап работы, который включает в себя обучение нескольких классификаторов и сравнительное тестирование качества, по результатам которых выбирается та модель, что продемонстрирует наибольшую точность.

Из текстов БрУК были выбраны две категории: «*A*» — «Пресса» и «*I*» — «Художественная литература». Категории были выбраны с позиции количества текстов, так как остальные категории содержат очень малое количество текстов и не могут полностью описать свою категорию. Все выбранные текстовые документы были разделены случайным образом на два равных набора, с сохранением примерно равного количества ресурсов по классам.

Обучение проводилось на одном из этих двух наборов (обучающая выборка), а тестирование — на другом. Далее все текстовые документы из обучающей выборки поделили на пять частей. Изымая первую часть документов из обучающей выборки, обучение классифика-




```
cv_df.groupby('model_name').accuracy.mean()

model_name
LinearSVC                 0.806193
LogisticRegression        0.737551
MultinomialNB             0.731942
RandomForestClassifier    0.685755
Name: accuracy, dtype: float64
```

Рис. 8. Результаты построенных классификаторов

тора проводилось на оставшихся документах. Используя тестовую выборку, определялись показатели качества работы классификатора. Затем изымая вторую часть из обучающей выборки, вычислялись другие значения показателей качества работы и т.д. В итоге было получено пять значений показателей качества работы классификатора. После этого наборы менялись местами и прогоны повторялись. В качестве окончательных результатов бралось их среднее арифметическое значение. Данное усреднение позволило сгладить результаты — тем самым сделать их более объективными.

Листинг для построения классификаторов предоставлен на рис. 7.Итак, лучшие результаты показала модель на основе метода опорных векторов. Ее средняя точность — 0,80 (рис. 8).

Заключение

Так как в качестве модели текста принималась цифровая модель *TF-IDF*, то особенностей в результатах классификации украиноязычных текстов от классификации текстов на других языках нет.

Отличия встречаются при предварительной обработке текста, когда приводим текст к цифровому виду. Поскольку украинский язык относится к флективным языкам, то при предварительной обработке украинских текстов, можно применять алгоритмы стемминга, суть которых — приведение слов к начальным словоформам путем отбрасывания аффиксов. Алгоритмы стемминга работают значительно быстрее, чем алгоритмы лемматизации. Но сегодня для украинского языка такие алгоритмы не наработаны.

Следует отметить, что данные результаты получены с учетом специфики БрУК, в которой определены наибольшие коэффициенты для первой и последней категории текстов. Учитывая это, уместно дополнить корпус текстов так, чтобы каждый класс имел примерно одинаковое количество материалов.

В будущем возможно применить нейронные сети или ансамбли моделей. Поскольку для последних специалисты рекомендуют как можно больший спектр данных — пополнение корпуса является приоритетной задачей.

REFERENCES

1. Alekseenko, L.A., Darchuk, N.P., Zuban, O.M., 2001. "Methodology for the Creation of the Automated System of Morpheme-Word Formation Analysis (ACMSA) of the words of the Ukrainian language". Scientific heritage of Prof. S.V. Semchynsky. Collection of scientific works. K., part 1, pp. 38–49. (In Ukrainian).
2. Brownian Corps of the Ukrainian Language, https://github.com/brown-uk/corpus/ (In Ukrainian).
3. Large electronic dictionary of the Ukrainian language, https://github.com/brown-uk/dict_uk/ (In Ukrainian).
5. LanguageTool API NLP UK, https://github.com/brown-uk/nlp_uk/
6. Starko, V.F., 2014. "Formation of the Brownian Corps of the Ukrainian Language. Linguistic and conceptual pictures of the world", 48, pp. 415–421. (In Ukrainian).
7. Starko, VF., 2013. "Classical Approach to Categorization in Language-Cognitive Studies". Linguistic and conceptual pictures of the world, 43(4), pp. 117–123. (In Ukrainian).
8. Jones, K.S., 2004. "A statistical interpretation of the term specificity and its application in retrieval". Journal of Documentation. MCB University Press, 60 (5), pp. 493–502.
9. Pirotte, F., Sunar, F., Piragnolo, M., 2016. "Benchmark of machine learning methods for the classification of a Sentinel-2 image". Int. Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences, 41, pp. 335–340.
10. Press William, H., Teukolsky, Saul A., Vetterling William, T., Flannery, Brian P., 2007. Section 16.5. Support Vector Machines. Numerical Recipes: The Art of Scientific Computing (View 3rd). New York: Cambridge University Press.
11. Russell Stuart, Norvig Peter, 2003. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall
12. Omid's Logistic Regression tutorial. http://www.omidrouhani.com/research/logisticregression/html/logisticregression.htm

Received 15.01.2019

E.A. Bobrovnyk, Master of the Department of Ukrainian Language and Applied Linguistics Institute of Philology of Taras Shevchenko National University of Kyiv, boulevard Taras Shevchenko, 14, Kyiv, 01601, Ukraine, mailkatherine.bobrovnik@gmail.com

K.K. Dukhnovska, Assistant of the Department of Applied Information Technologies of the Faculty of Information Technologies of Taras Shevchenko National University of Kyiv Glushkov ave., 4, Kyiv, 03022, Ukraine, duchnov@ukr.net

N.V. Piroh, Assistant of the Department of Applied Information Technologies of the Faculty of Information Technologies of Taras Shevchenko National University of Kyiv Glushkov ave., 4, Kyiv, 03022, Ukraine, mykola.pyroh@ukr.net

THEMATIC CLASSIFICATION OF UKRAINIAN TEXTS, DIFFICULTIES OF ITS INTRODUCTIONS

Introduction. One of the major tasks of artificial intelligence, information search, and word processing is that of classification. The main problem with the automated classification of text information is that a document is presented in ahuman language and does not belong to structured data.

Solutions representing the classification of English and Russian texts are numerous. But, no research describing algorithms of developing a classifier for the texts written in the Ukrainian language as well as their peculiarities has been found by the authors.

Purpose Specify the peculiarities of the automated classification of texts written in the Ukrainian language.

Results. BrUC is the only *corpus* of Ukrainian texts on open access, the texts of which can be used to develop algorithms and methods of classification of texts in the Ukrainian language.

To develop classifiers of Ukrainian texts the following methods and algorithms have been used: *Random Forest Classifier, Support Vector Machines, Naive Bayes Classifier,* and *Logistic Regression*. Supervised learning is used for training all these classifiers. The essence of the method is that a ready-made set of classified texts represented by BrUC is used for learning.

Conclusions. The model for classification of Ukrainian texts on the basis of support vector machines has demonstrated the best results. Its mean accuracy is 0,80.

Keywords*: классификатор текстовых документов, корпус документов, метод контролированного обучения.*

К.А. Бобровник, магістрант, кафедра укр. мови та прикладної лінгвістики,
Ін-т філології Київського національного університету ім. Тараса Шевченка,
бульвар Тараса Шевченка, 14, Київ 01601, Україна,
mailkatherine.bobrovnik@gmail.com

К.К. Духновська, асистент, кафедра прикладних інформаційних технологій,
Київський національний університет ім. Тараса Шевченка,
просп. Академіка Глушкова, 4, Київ 03022, Україна,
duchnov@ukr.net

М.В. Пирог, асистент, кафедра прикладних інформаційних технологій,
Київський національний університет ім. Тараса Шевченка,
просп. Академіка Глушкова, 4, Київ 03022, Україна,
mykola.pyroh@ukr.net

ТЕМАТИЧНА КЛАСИФІКАЦІЯ УКРАЇНСЬКОМОВНИХ ТЕКСТІВ, ТРУДНОЩІ ЇЇ ВПРОВАДЖЕННЯ

Вступ. Однією з важливих задач штучного інтелекту, інформаційного пошуку та систем текстової обробки є задача класифікації. Головна складність автоматичної класифікації текстової інформації в тому, що документ представлений на природній мові і він не відноситься до структурованих даних.

Рішень для класифікації англомовних чи російськомовних текстів багато. Проте робіт, в яких описано алгоритми побудови класифікатора для текстів, поданих на українській мові, та в чому їх особливість автори не знайшли.

Мета роботи — визначити особливості автоматичної класифікації текстів, поданих на українській мові.

Результати. БрУК — єдиний корпус українських текстів у відкритому доступі, тексти якого можна використовувати для розробки алгоритмів і методів класифікації українськомовних текстів.

Для побудови класифікаторів українськомовних текстів використовувались такі методи та алгоритми: *Random Forest Classifier, Support Vector Machines, Naive Bayes Classifier.* и *Logistic Regression*. Метод контрольованого навчання використовується для тренування всіх цих класифікаторів. Суть цього методу в тому, що для навчання використовується вже готовий класифікований набір текстів, яким є БрУК.

Висновок. Кращі результати показала модель класифікації українськомовних текстів на основі методу опорних векторів. Її середня точність — 0,80.

Ключові слова*: класифікатор текстових документів, корпус документів, метод контрольованого навчання.*