Method Of Recognition Sarcasm In English Communication With The Application Of Information Technologies

Serhii Trystan[1], Olha Matiushchenko[2] , Maryna Naumenko [3]

[1]*Kharkiv National University of Radio Electronics, Nauky Ave. 14, Kharkiv, 61166, Ukraine*
[2,3]*Ivan Kozhedub Kharkiv National Air Force University, Sumska str. 77/79, Kharkiv, 61023, Ukraine*

Abstract

The article developed a software application for recognizing sarcasm in English communication. NLP technology is used to implement machine learning. Python programming language. Comparisons with known algorithms and models are made. The advantage in the simplicity of the method implementation and the speed of recognition, which corresponds to live communication, is proved.

Keywords

Artificial intelligence, language recognition, machine learning, sarcasm.

1. Introduction

Human language is extremely complex and contains a significant number of linguistic constructions. Language recognition and translation is a well-developed area of machine learning. However, living human language contains such elements as humor, irony, pun, aphorism, sacredness, which are not always correctly recognized by native speakers, and recognizing them with the help of intelligent information technology becomes quite a difficult task. At the same time, virtual translators must, in a reasonable amount of time (preferably in real time), recognize a person's living language and communicate its content and emotional and logical implication to the user. Today, the universal language of communication is English. Sarcasm is the most complex language construction, because in a sentence with sarcasm one logical construction is confirmed, and the opposite is understood [1]. Thus, recognizing sarcasm in communication is quite a challenge.

2. Analysis of problem-solving methods

At present, there are a sufficient number of electronic translators designed to facilitate the communication process.

Living language recognition is based on electronic dictionaries and Data Science (DS) technologies [2]. In DS such direction as Nature - Language Processing (NLP) is allocated. This area studies the problems of computer analysis and synthesis of natural language.

For artificial intelligence, analysis means understanding language, and synthesis means generating intelligent text [3].

Solving the problems associated with the analysis and synthesis of language structures will mean creating a more convenient form of interaction between computer and human, as well as ensuring communication through electronic translators.

Examples of using NLP are such services and applications as Siri (assistant for operating systems from Apple: iOS, watchOS, macOS, HomePod and tvOS) [4], Cortana (virtual assistant in Windows) [5], Gmail Spam Filter

(analysis service) and selection of mail with spam) [6].

It should be noted that there are currently a sufficient number of applications that implement NLP.

III International Scientific And Practical Conference "Information Security And Information Technologies", September 13–19, 2021, Odesa, Ukraine
EMAIL: serhii.trystan@nure.ua (A. 1); mog28@ukr.net (A. 2); mv.naumenko@ukr.net (A. 3)
ORCID: 0000-0003-1270-5254 (A. 1); 0000-0003-4843-8258 (A. 2); 0000-0002-1216-9263 (A. 3)

However, simple solutions are needed that will quickly recognize sarcasm in living language in communication.

The theoretical basis of the work was works [7], [8] works on applied statistical analysis [9] and applied analysis of text data in Python [10].

3. Main part

3.1. Problem statement and substantiation of tools for its solution

The task to be solved in the article is the automatic recognition of sarcasm in live English communication.

The main tools for solving this problem are:

1. Statistical and mathematical methods of big data processing [7]–[10];
2. Dataset for model learning [11];
3. Python programming language 3.8.1 [12];
4. Jupyter notebook development environment;
5. A set of libraries (sklearn, re, pandas, numpy, nltk (natural language toolkit), matplotlib).

3.2. Dataset choosing and bring it to normal

The date set was chosen on Kaggle.com. The required data set is called “News Headlines Dataset For Sarcasm Detection. High quality dataset for the task of Sarcasm Detection” [11]. This dataset contains news headlines that are collected from two sites: TheOnion and HuffPost. Each record consists of three attributes, and itself:

1. **is_sarcastic** (1, if the entry is sarcastic, and 0 if not sarcastic);
2. **headline** (the title of the page);
3. **article_link** (link to the page from which the title was taken).

In Figure 1 shows the structure of this dataset.

	article_link	headline	is_sarcastic
0	https://www.huffingtonpost.com/entry/versace-b...	former versace store clerk sues over secret 'b...	0
1	https://www.huffingtonpost.com/entry/roseanne-...	the 'roseanne' revival catches up to our thorn...	0
2	https://local.theonion.com/mom-starting-to-fea...	mom starting to fear son's web series closest ...	1
3	https://politics.theonion.com/boehner-just-wan...	boehner just wants wife to listen, not come up...	1
4	https://www.huffingtonpost.com/entry/jk-rowlin...	j.k. rowling wishes snape happy birthday in th...	0
...	...	...	...
26704	https://www.huffingtonpost.com/entry/american-...	american politics in moral free-fall	0
26705	https://www.huffingtonpost.com/entry/americas-...	america's best 20 hikes	0
26706	https://www.huffingtonpost.com/entry/reparatio...	reparations and obama	0
26707	https://www.huffingtonpost.com/entry/israeli-b...	israeli ban targeting boycott supporters raise...	0
26708	https://www.huffingtonpost.com/entry/gourmet-g...	gourmet gifts for the foodie 2014	0

26709 rows × 3 columns

Figure 1: “Sarcasm_Headlines_Dataset” structure

As can be seen from fig. 1, the first steps in creating an information technology for sarcasm recognition - is to bring the column "headline" to normal, which consists of bringing all the letters to lowercase, removing dots and spaces.

In Figure 2 shows the script for bringing the “headline” column to normal.

```
In [7]: ds = pd.read_json('Sarcasm_Headlines_Dataset.json', lines = True)
        headline_re_sub = []
        for i in ds['headline']:
            headline_re_sub.append(re.sub('[^a-zA-Z]', ' ',i))
        ds['headline'] = headline_re_sub
```

Figure 2: Bring the column "headline" to normal

3.3. Stemming words

The next stage of the method is word stemming. In the field of natural language processing, there are cases when two or more words have a common root. Stemming reduces all counter word forms to one, normal vocabulary form.

There are two main steaming algorithms: Porter's algorithm and Lancaster's algorithm [9]. The developed script uses Porter's algorithm because it is less aggressive to word forms. Lancaster's algorithm is quite aggressive, because it strictly "cuts" the word and makes it very confusing, which is impractical in recognizing such a complex linguistic phenomenon as sarcasm. In Figure 3 shows the use of the Portrait algorithm with respect to the "headline" column.

```
In [29]: headline_re_sub =[]
         for i in ds['headline']:
             headline_re_sub.append(re.sub('[^a-zA-Z]', ' ',i))

         ps = PorterStemmer()

         ds['headline'] = headline_re_sub

         features = ds['headline']
         labels = ds['is_sarcastic']

         features = features.apply(lambda x: x.split())
         features = features.apply(lambda x : ' '.join([ps.stem(word) for word in x]))

         tv = TfidfVectorizer(max_features = 5000)

         features = list(features)
         features = tv.fit_transform(features).toarray()
         features

Out[29]: array([[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]])
```

Figure 3: Using Porter's algorithm for word stemming in a dataset

3.4. Convert text to numbers

The next step of the method is to convert the text into a meaningful representation of numbers, which will be used in machine learning algorithms for prediction. In Figure 4 shows the use of the TfidfVectorizer function, which was taken from the sklearn.feature_extraction.text library.

```
In [28]: headline_re_sub =[]
         for i in ds['headline']:
             headline_re_sub.append(re.sub('[^a-zA-Z]', ' ',i))



         ds['headline'] = headline_re_sub

         features = ds['headline']
         labels = ds['is_sarcastic']

         ps = PorterStemmer()

         features = features.apply(lambda x: x.split())
         features = features.apply(lambda x : ' '.join([ps.stem(word) for word in x]))
         features

Out[28]: 0        former versac store clerk sue over secret blac...
         1        the roseann reviv catch up to our thorni polit...
         2        mom start to fear son s web seri closest thing...
         3        boehner just want wife to listen not come up w...
         4        j k rowl wish snape happi birthday in the most...
                                      ...
         26704                  american polit in moral free fall
         26705                              america s best hike
         26706                                  repar and obama
         26707    isra ban target boycott support rais alarm abroad
         26708                          gourmet gift for the foodi
         Name: headline, Length: 26709, dtype: object
```

Figure 4: Using the "TfidfVectorizer" function

After all the steps to bring the data to values that can be used in computer training, we need to determine the model of machine learning.

3.5. Choosing a machine learning model

The logistic regression is chosen as the basic model of machine learning. Logistic regression is a machine learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable containing data encoded as 1 (yes, success) or 0 (no, failure). Since our problem is a binary classification problem (1 - sarcasm, 0 - not sarcasm), logistic regression is a relevant model [9].

In Figure 5 presents a graph of logistic regression.

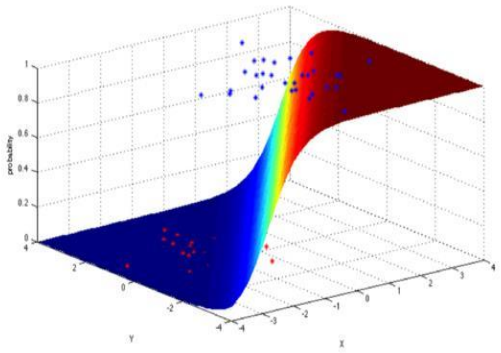

Figure 5: Logistic Regression Plot

3.6. Machine learning

Before training the model, divide the dataset into training and test samples with the following parameters: 30 percent of the entire sample will go to the test data set, and the other 70 to the training (Figure **6**).

```
X_train, x_test, Y_train, y_test = train_test_split(features, labels, test_size = .30, random_state = None)
```

Figure 6: Division of the dataset into training and test samples

The accuracy of the model is checked using cross-validation (re-sampling procedure). The decision to choose this method is based on its simplicity and obtaining a less biased or less optimistic assessment of the quality of the model than other methods. In

Figure **7** shows the accuracy of the model that was verified by cross-validation.

```
model = LogisticRegression()
model.fit(X_train, Y_train)
cv = 10
res = cross_val_score(estimator = model, X = X_train, y = Y_train, cv = cv)
print("The average quality of the model, when conducting ",cv," experiments: ",round(np.mean(res),2))
print("Values in the training sample: ", round(model.score(X_train, Y_train),2))
print("Values in the test sample: ", round(model.score(x_test, y_test),2))
```

```
The average quality of the model, when conducting  10  experiments:  0.83
Values in the training sample:  0.88
Values in the test sample:  0.83
```

Figure 7: Model accuracy

Another metric for evaluating the quality of the model is the ROC curve (one of the most popular quality functionalities in binary classification problems) [9].

In Figure 8 shows the ROC - curve obtained in the work.

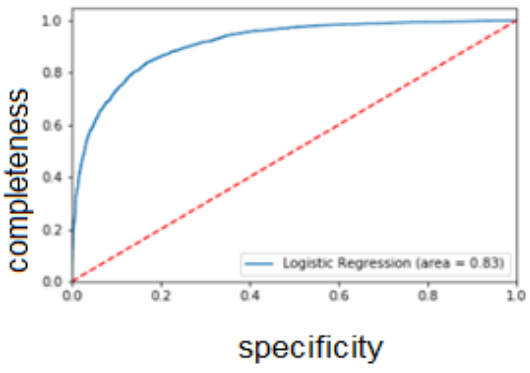

Figure 8: ROC curve

After carrying out stages designing model of machine learning it is necessary to carry out the test.

3.7. Model testing

In Figure 9 shows an example of testing the model on the phrase: *"Oh, have I touched that tiny ego of yours?"*.

Information technology has rightly determined that this sentence is sarcasm.

```
sent = input(str())
sent = re.sub('[^a-zA-Z]', ' ', sent)
text = []
text.append(sent)
data = tv.transform(text).toarray()
pred = model.predict(data)
if(int(pred[0])==1):
    print("Sarcasm")
else:
    print("NOT Sarcasm")
```

```
Oh, have I touched that tiny ego of yours?
Sarcasm
```

Figure 9: Model testing

It is also necessary to compare the results with known algorithms and implementations. This comparison is shown in Table 1.

Table 1
Comparison of the obtained results

Model	KNearest Neighbors	**Logistic Regression**	Naive Bayes	LinearSVC	RandomForest Classifier
Prediction	68.3% (0,8 sec)	**84.04% (0,12 sec)**	75.09% (0,10 sec)	78.08% (0,11 sec)	79.64% (0,11 sec)

Thus, we can conclude about the effectiveness of the developed method and the possibility of its application for the recognition of sarcasm in live communication on English.

4. Conclusions

1. Recognition of linguistic constructions of natural language is a difficult task on the border of such scientific areas as AI, ML, philology.
2. Recognizing sarcasm in living language is one of the most difficult tasks, because a person masks sarcasm.
3. Recognition of sarcasm should be done quickly (commensurate with the pace of communication), so decisions should be simple and effective.
4. The obtained solution has a degree of recognition of 0.83, but in contrast to more powerful solutions, it is quite fast.
5. The article presents the results of the study of ML and NLP in terms of solving the problem of identification and classification of sarcasm.

5. References

[1] "Definition of SARCASM" [Online]. Available: www.merriam-webster.com. [Accessed 29 June 2021].

[2] "About Data Science / Data Science Association" [Online]. Available: www.datascienceassn.org. [Accessed 3 April 2021].

[3] Goldberg Y. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 2016, No *57*, pp. 345-420.

[4] Bhat H. R., Lone T. A., Paul Z. M. Cortana-intelligent personal digital assistant: a review. *International Journal of Advanced Research in Computer Science*, 2017, No *8*(7), pp. 55-57.

[5] Fumera G., Pillai I., Roli F. Spam filtering based on the analysis of text information embedded into images. *Journal of Machine Learning Research*, 2006, No. 7 (12), pp. 2699-2720.

[6] Liddy, E.D. 2001. Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc.

[7] Practical statistics for Data Science specialists: Translated from English / P. Bryus, E. Bryus. Petersburg, 2018. 304 p.

[8] Bengfort B., Bilbro R., Okheda T., Applied Text Data Analysis in Python. Machine Learning and the Creation of Natural Language Processing Applications, St. Petersburg: Peter, 2020.368 p.

[9] Atienza R. Advanced Deep Learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more. Packt Publishing Ltd, 2020. p. 512.
[10] "News Headlines Dataset For Sarcasm Detection" [Online]. Available: https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection. [Accessed 29 June 2021].
[11] Guttag J. V. Introduction to Computation and Programming Using Python: With Application to Understanding Data, 2017. p. 447.
[12] Oleksandr Laptiev, Savchenko Vitalii, Serhii Yevseiev, Halyna Haidur, Sergii Gakhov, Spartak Hohoniants. The new method for detecting signals of means of covert obtaining information. 2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (IEEE ATIT 2020) Conference Proceedings Kyiv, Ukraine, November 25-27. pp.176 –181.
[13] Oleg Barabash, Oleksandr Laptiev, Valentyn Sobchuk, Ivanna Salanda, Yulia Melnychuk, Valerii Lishchyna. Comprehensive Methods of Evaluation of Distance Learning System Functioning. International Journal of Computer Network and Information Security (IJCNIS). Vol. 13, No. 3, Jun. 2021. pp.62-71, DOI: 10.5815/ijcnis.2021.03.06.