Advances in Clickbait and Fake News Detection Using New Language-independent Strategies

Claudia Ioana Coste and Darius Bufnea

Original scientific article

***Abstract*—Online publishers rely on different techniques to trap web visitors, clickbait being one such technique. Besides being a bad habit, clickbait is also a strong indicator for fake news spreading. Its presence in online media leads to an overall bad browsing experience for the web consumer. Recently, big players on the Internet scene, such as search engines and social networks, have turned their attention towards this negative phenomenon that is increasingly present in our everyday browsing experience. The research community has also joined in this effort, a broad band of detection techniques being developed. These techniques are usually based on intelligent classifiers, for which feature selection is of great importance. The work presented in this paper brings our own contributions to the field of clickbait detection. We present a new language-independent strategy for clickbait detection that takes into consideration only features that are general enough to be independent of any particular language. The methods presented in this paper could be applied to web content written in different languages. In addition, we present the results of a complex experiment that we performed to evaluate our proposed method and we compare our results with the most relevant results previously obtained in the field.**

***Index Terms*—clickbait detection, features, intelligent classifier, natural language, accuracy.**

I. Introduction

Before the Internet age, the old media industry relied on printed materials such as newspapers or magazines, television being the most advanced form of media, technologically speaking, at that time. As the Internet grew and its usage by the masses increased, the media industry was put under a constant pressure to revolutionise its business model. A major shift has been observed: classical content providers had to switch to online editions to survive, their printed editions decreasing constantly over time. New concepts and topics were born: user interaction and engagement, online subscribers, content sharing through different social networks, ads targeting, social media. In this new ecosystem, most websites do not charge web visitors a subscription for their provided content. Instead, to realise their income, they rely on different monetization techniques such as affiliate marketing or advertisement display. To raise their advertising revenue, content providers need also to increase their page views. To achieve this, the web visitor has to stay trapped as much as possible in the content providers' networks, different deceiving techniques being used for this purpose. Clickbait is one of these bad habits content providers are known for. Unfortunately, the media industry today relies very often on such deceptive methods with a negative impact on the overall user satisfaction and experience.

The term clickbait is in general associated with links to articles that have misleading, incomplete, or confusing titles. Sometimes such links exaggerate the actual content they link to or they point out a shocking title to the web visitor. The term itself was born in the first part of the 20th century, when the TV audience was advised not to change the channel during the commercial breaks, because they "wouldn't believe what happens next". More recently, according to the Oxford Advanced Learner's Dictionary, this word refers to "material put on the Internet in order to attract attention and encourage visitors to click on a link to a particular web page". Even though relying on such a technique defies the basic codes of journalistic standards and ethics, it is a common practice among content providers to abusively use it to increase their page views and advertising revenue. By clicking on such a link, users will have, most of the time, a bad user experience. An incomplete or confusing text link, sometimes over exaggerated by the publisher, leaves the web visitors frustrated or disappointed because the reached article does not meet their expectations (as suggested by the text or image link).

There are certain situations when a clickbait is triggered outside its publisher network: a catchy or incomplete title can mislead a web visitor on third party websites that are not directly affiliated with the bait's publisher. For example, a bait link can propagate in a search engine result page (SERP) [1], in a social network news feed or, when shared, in users' posts in social media. All of these third party websites may be impacted by the user's unhappy and time-consuming experience.

Writing bait articles implies, most of the time, more effort put into deceiving the web visitor rather than into creating high-quality content. Such articles usually exploit a human vulnerability known as the curiosity gap. Types of articles that are shared the most through clickbait links include, but are not limited to: gossip, unfounded rumours, fake news or, any other type of thin content article. Authors of [2] studied the relation between fake news and clickbait links. They stated that the appearance of such a link is sometimes associated with false (or inaccurate) information and by using such links, fake news

Manuscript received February 19, 2021; revised July 8, 2021. Date of publication August 30, 2021. Date of current version August 30, 2021. The associate editor prof. Ana Sović Kržić has been coordinating the review of this manuscript and approved it for publication.

The part of this paper was presented at the International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2020).

Claudia Ioana Coste is a master student in Distributed Systems in Internet, Computer Science Department, Babeş-Bolyai University, Cluj-Napoca, Romania (e-mail: c.i.coste@cs.ubbcluj.ro)

Darius Bufnea is a lecturer professor within the Computer Science Department, Babeş-Bolyai University, Cluj-Napoca, Romania (e-mail: darius.bufnea@ubbcluj.ro).

Digital Object Identifier (DOI): 10.24138/jcomss-2021-0038

is widely spread. Gilbert et al. revealed in their study [3] that once a false perception has been formed in someone's brain, a correction of that perception is very unlikely. Baruch Spinoza's principles state that the human brain inclines to classify in the very first step as true any information received from the senses, and later on, in a second step it starts questioning about how valid the received information is. Very frequently, this second step is omitted due to internal or external factors such as: source's credibility, stress, noise, or tiredness. Web publishers exploit and rely on this specific "human" vulnerability to make use of a clickbait.

Considering the aforementioned arguments, it is essential to investigate this negative phenomenon and take measures against it. As news consumers have become lately increasingly impacted by the existence of clickbait, low quality articles, or thin content, the research community turned its attention towards these negative aspects presented in the web space, with efforts in offering users a more pleasant and less deceiving browsing experience. The current paper aims to bring its own contributions to this challenge. We present an overview of the most important clickbait detection methods developed so far by the research community. In addition, we introduce a new language-independent strategy for clickbait detection. To evaluate this strategy, we compare the results of the performed experiments with the most relevant results presented previously in the literature. This paper is an extended work of the research presented in [4]. Additionally to the work presented in [4], we added some new characteristics for the intelligent model: subjectivity, polarity, number of negative/positive words, N-grams and tested our new model with different configurations using other machine learning algorithms besides Random Forest: Naive Bayes, Logistic Regression, Support Vector Machine and Decision Tree.

The rest of this paper is organized as follows. The next section presents the main results of other related work in the field. Section III describes our new proposed strategy, and the subsequent results obtained through the experiments; the comparative analysis of the results is being done in Section IV. Section V concludes our paper, summarizing our results and presenting possible future work.

II. Related Work

As we previously mentioned, as this negative phenomenon is increasingly present in online media, it got into the attention of the research community but also of big players on the WWW scene such as search engines or social networks. Most of the studies that have been performed so far to identify clickbait links and the corresponding articles are using machine learning based techniques.

As part of a relatively young field of study, the clickbait detection methods developed so far are not very accurate (less than 95% accuracy). Another drawback in the field is that all available datasets used for training and testing different intelligent classifiers contain exclusively English records. Because a clickbait link might be present in articles written in any language, it is of great importance to train and test the developed methods on more than one language.

Efforts in this sense have been made by the authors of [1]. They developed a Chrome browser plugin that allows users to report links considered as clickbait. Based on the users' reports, a multilingual community-driven clickbait database is being currently built. Authors of [5] continued the work presented in [1] and developed a method for filling up a sample database with non-clickbait entries extracted from users' navigational paths logged in a proxy server log. Having both types of samples (either classified as clickbait, either as non-clickbait) is vital for successful training of different machine learning algorithms.

In [6], Biyani et al. classified clickbaits into the following categories (a clickbait may fall into more than one category):

- confusing titles;
- articles containing erroneous information;
- punctuation formatting (overuse of uppercases, question marks, and exclamation marks);
- details omitting;
- bait and switch (users have to perform an additional click to get the full content – this is in order not to increase a variable called bounce rate for the bait website);
- exaggeration;
- reporting of vulgar and unbelievable stories;
- usage of vulgar words.

Authors of the aforementioned study used decision trees to develop a classification algorithm that takes into consideration the following attributes: URL, title, similarity between title and content, use of references, and informality. They obtained an accuracy of over 74%, the use of references and informality being the most relevant features. To benefit from users' curiosity gap, content providers often rely on references by making use of third person pronouns, demonstratives (determiners or pronouns that point to particular nouns), definite articles or by using adverbs at the beginning of the title.

Authors of [7] based their research on a browser plugin that they developed and that automatically detects clickbaits. At the same time, their plugin offers an option to blacklist such links. To detect clickbait links, a classification algorithm was implemented. This algorithm takes into consideration only the titles, uses supervised learning models and 10-fold validation and it relies on a natural language processor (Stanford NLPCore [8]). The achieved accuracy was 93%. The used clickbait samples were taken from publications already known for their thin content articles, while the non-clickbait samples link to more than 18000 high-quality Wikinews articles. Among the most important conclusions study [7] reached are that clickbait titles use personal or possessives pronouns or causal complements, and adverbs, while proper nouns are more specific to high-quality articles. Another conclusion was that non-clickbait and clickbait articles tend to use verbs in two different ways to form a proper phrase. High quality articles use participles and third person, singular, while clickbait articles usually use past tense verbs in the first and second person.

The study presented in paper [9] proposed a new model for the detection of clickbait, model based on over two hundred features. The authors of this study compiled one of the first clickbait corpuses of almost 3000 Twitter tweets, each of them manually classified by a human volunteer as being clickbait

or not. These tweets were gathered from different online publishers that have a high number of readers such as New York Times, CNN, BBC News, Fox News or The Guardian. The proposed model took into consideration three sources for extracting information: the clickbait link, the metadata of the publisher's website, and the bait message itself. Furthermore, the bait message was analyzed from three different perspectives: dictionary categories (usage of specific words and expressions), language patterns, and textual statistics. The best accuracy obtained for a RFC (Random Forest Classifier, [10]) was 79%, but, as we previously mentioned, the main problem and disadvantage of the proposed corpus is that it contains only English language entries, the model being trained only on these types of records.

Authors of the paper [11] organized a global contest called Clickbait Challenge, whose main purpose was to draw attention about this negative phenomenon and at the same time to find new ways to fight against it. For this contest, they provided a database containing clickbait and non-clickbait samples and populated with Twitter tweets. Each sample was annotated in a regression manner by volunteers: 1 means clickbait, 0.66 quite a clickbait, 0.33 slightly a clickbait and 0 means not a clickbait at all. This database is still available to researchers who can submit new solutions and proposals for validation. The best ranking algorithms in the Clickbait Challenge are presented in table I.

In [16], the authors analyze some relatively recent major media and political events such as Brexit referendum and Trump's election in the USA from a different perspective. In both these events, fake news and clickbait were considered as having a substantial role, paper [16] discussing their characteristics but also their ideological and financial implications.

Article [17] approach is to treat the clickbait problem as a regression one to predict the intensity of the clickbait. The dataset used was the one provided by The Clickbait Challenge, and they are proud to say that their method is the best compared to The Clickbait Challenge results. Their approach consists of multiple regression models (Simple Linear Regression, Ridge Regression, Gradient Boosted Regression, Random Forest Regression, Adaboost Regression) combined with word and sentence embedding representations and transformer representations. The best approach obtains over 87% accuracy and a MSE of 0.02.

Paper [14] describes a solution named Zingel of The Clickbait Challenge, ranked first in 2017. Authors reformulate the regression clickbait problem as a multiclassification one, based on the 4-point annotation scale (0 - not clickbait, 0.33 - slightly clickbait, 0.66 - considerably clickbait, 1 - heavily clickbait). They used self-attentive neural networks with the hidden states generated by bidirectional Gated Recurrent Units (biGRU), which allows the developers to completely train the network without any manual feature engineering.

Authors of paper [18] propose a clickbait convolutional neural network (CBCNN) to outperform traditional clickbait detection methods that require a lot of work for feature engineering. They apply CBCNN on a pretrained Word2Vec, also taking into consideration specific features from the articles analyzed. Word2Vec is a neural network technique used to convert words into numbers, taking into consideration the words dynamic and associations [19]. By combining CNN and Word2Vec, they succeeded in outperforming other five baselines, considering automatically computed features such as: word sequence, word meaning, and writing style of each article.

III. Clickbait Detection Strategy

In this paper, we advance a method for clickbait detection that makes use of some artificial intelligent-based classifiers and relies only on features that are language independent. Considering this, our method can be trained and applied for classifying articles whatever the language they are written in. Authors of [20] followed the same goal, their approach being based on CNN models (Convolutional Neural Networks) that use features such as distributed word or character embeddings. The main steps of our approach will be presented next, describing each decision taken in our clickbait detection strategy. At the same time, we evaluate the results of our performed experiments to compare them with other results from the literature.

Our clickbait detection strategy is based on a set of decisions related to the classification model, application of an intelligent classifier, finding the best algorithm and the best metrics of accuracy.

If we consider only language-independent features for this intelligent model, we can easily apply it on other sample articles in multiple languages or even on articles written in more than just one language. We need to mention that the proposed model needs to be trained also on other languages in order to work in a multilanguage environment.

A diagram of our project methodology can be seen in Fig. 1, where we split our research into eight main steps (colored in green). The selected features were put into four categories (presented in orange), while the steps used to extract lexical features are presented in grey. In the beginning, we analyzed the features and datasets used in literature, and then we chose the most relevant dataset for our purpose. Next, we preprocessed the data, excluding records with noise, and then we extracted language-independent characteristics (orthographic, grammatical, lexical, and text metrics). Before using the extracted features within an intelligent classifier, we normalized the data to fit into the [0,1] interval. The considered intelligent algorithms (naive Bayes, logistic regression, decision tree, random forest, and support vector machine) were chosen based on the best results obtained previously in literature studies. The final steps were: calibrating the parameters of the intelligent models and computing the metrics, comparing also our results with what was obtained before in literature.

We will discuss each of the steps presented in the aforementioned methodology.

Analyzing clickbait literature. By analyzing relevant articles about clickbait detection, we came to the conclusion that the most relevant characteristics were extracted from the title of the article and the post text (the bait message that lures readers into clicking the link). In quite a few articles, we came across features selected from the metainformation of the clicked website [9] and from the targeted web page itself [6].

TABLE I
THE CLICKBAIT CHALLENGE RANKING

Rank	Algorithm	Accuracy	Precision	Relies on
1st	"albacore" [12]	85.5%	73.1%	a recursive bidirectional neural network of type biGRU [13]
2nd	"zingel" [14]	85.6%	71.9%	recursive and bidirectional neural networks
3rd	"carpetshark" [15]	84.7%	72.8%	a support vector machine

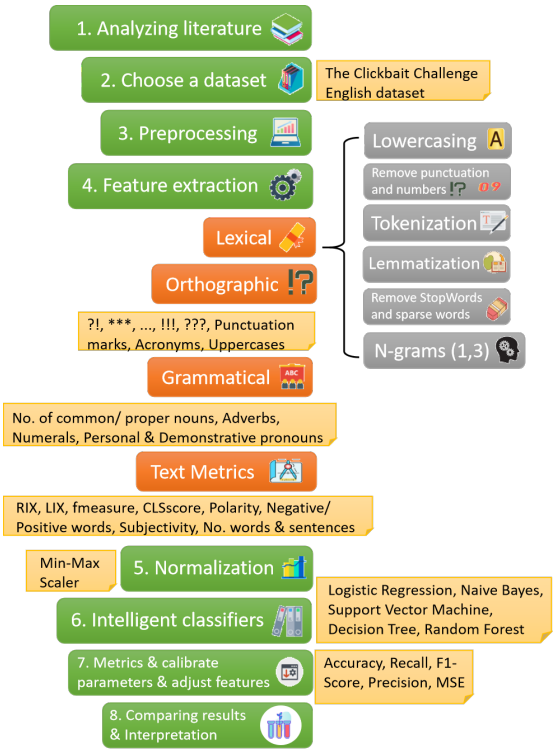

Fig. 1. Project's Methodology

The characteristics that have a high importance score in the clickbait literature are: N-grams, meaningful group of words, frequent punctuation patterns, formality measures ($fmeasure$ [21], LIX and RIX indexes [22], $CLScore$ [23]). Additionally, the number of proper or common nouns, uppercases, morphological or syntactic patterns obtained a high accuracy as well.

Choosing a dataset. We used clickbait and non-clickbait samples to train and test our chosen intelligent classifiers. The dataset used was the one provided by The Clickbait Challenge [24]. The dataset records were taken from news accounts and well-known media outlets through the Twitter API. The sample annotation was done by five different volunteers, each of them completing several questionnaires created on the Amazon Mechanical Turk Platform (AMT - https://www.mturk.com). Clickbait is not an objective phenomenon, but rather influenced by people's opinions, values, and emotions, as well as their cultural background as it is presented in [11]. Therefore, the samples were annotated with four classes: 0 (not clickbait), 0.33 (slightly clickbait), 0.66 (considerably clickbait), and 1.0 (heavily clickbait). Each labeled sample was verified and the suspicious ones were dropped out, resubmitting the associated questionnaire in the AMT platform.

The Clickbait Challenge provides three datasets: one unlabelled and two labelled. We chose to use one of the annotated datasets that is composed of two files in JSONL format and an images folder for the tweets that have a picture attached. The JSONL files have the structure described in Table II.

Eliminate noise from the dataset. Before training and testing our intelligent approach on the dataset, we want to check if there are any errors infiltrated into the data. Firstly, we verified that if the "truthMean" is above 0.5 (the threshold value), then the "truthClass" should be "clickbait" and "non-clickbait" otherwise. In addition, we verified if the "postText" ("the bait message") or "targetTitle" (the title of the analyzed article) or the "targetParagraphs" (the content of the news) are not empty in the record sampled. To obtain relevant results with our intelligent models, we need to balance our dataset. In other words, we equalize the number of clickbait and non-clickbait records as it is presented in Table III. Next, we randomly split our balanced dataset into 80% training and 20% testing. Both sets (training and testing) contain an equal number of samples per each class.

Engineering language independent features. The used characteristics are extracted only from the article's title (the "targetTitle" attribute from the "instances.jsonl" file), from the teaser message ("postText" attribute) and from the text content of the article ("targetParagraphs" attribute). The features are collected using the Stanford NLPCore [25] natural language processor and different Python's libraries: textblob, scikit-learn, and nltk (Natural Language Toolkit). Textblob was used for counting positive and negative words and metrics such as subjectivity or polarity [26]. From the scikit-learn library [27] we used multiple packages such as TfidfVectorizer for feature extraction and MinMaxScaler for the normalisation step. The TfidfVectorizer was used to determine N-grams from text, while the Nltk package was used because of its corpus that supports multilanguage stopwords [28].

Stanford NLPCore is a machine learning tool helping users to extract linguistic features from the text. We use it to parse text into sentences, sentences into words, and to label words as part of speech tags or syntactic tags. Stanford NLPCore is available for a large variety of programming languages, also for Python, our programming language of choice. Moreover, this intelligent tool has over 53 modules that can be downloaded and trained with the model to annotate 53 different languages [29]. The base of this labelling tool is a pretrained neuronal model, on top of whom is added the language module, containing specific treebanks. Treebanks (trees resulting from parsed text) compute the syntactic and morphological structure of the sentences. In our approach, we focused, first, to identify the language of the sample and then to use the suitable language module.

TABLE II
THE CLICKBAIT CHALLENGE DATABASE STRUCTURE [24]

File	Attributes	Data type	Observations
instances.jsonl Contains the tweet samples and other information collected later prefixed with "target"	id	string	
	postTimestamp	datetime	the date on which the post was published
	postText	string array	the text post without any links, meaning the bait message
	postMedia	string array	relative path to the attached photos from the "media" folder
	targetTitle	string	the title of the shared article
	targetDescription	string	article's description tags
	targetKeywords	string	keywords separated by comma
	targetParagraphs	string array	all paragraphs of the web news
	targetCaptions	string array	all the descriptions of the article's attached pictures
truth.jsonl Contains the annotations of the 5 volunteers, the mean and the output class	id	string	
	truthJudgments	float number array	the annotated scores labeled by the 5 volunteers
	truthMean	float number	the arithmetic mean of the five scores from "truthJudgments"
	truthMedian	float number	it represents the middle value of the 5-element array sorted ascending or descending
	truthMode	0.0 or 1.0	codified output classes
	truthClass	string	"clickbait" or "non clickbait", the name of the output class

TABLE III
DATASET DISTRIBUTION

Database	No. clickbait	No. non clickbait	No. deleted samples ("noisy" data or empty fields)
The Clickbait Challenge	5523	16474	8852
Total number of correct samples	2963	10182	-
Number of samples used for detection	2963	2963	-
Training	2370	2370	-
Testing	593	593	-

When annotating morphological and syntactic words or group of words, we take into consideration the universal part of speech tags (POS tags) presented in [30].

We chose the features presented in Table IV as being the most relevant to be included in our classification algorithm. To classify the extracted characteristics, we split them into four categories: Grammatical, Orthographic, Text Measures and Lexical. The grammatical category contains 13 features, including: number of common/proper nouns, number of nouns with subject function, adverbs, numerals, personal pronouns, and demonstrative pronouns. Orthographic category refers to punctuation, acronyms and usage of uppercase. Therefore, this category has 19 features such as: different punctuation patterns, usage of parentheses, usage of question and exclamation marks, number of acronyms, no. of uppercases used. An acronym is considered an upper-cased word with no more than five characters. The text measures class includes all metrics applied on text: formality indexes (LIX, RIX, fmeasure, CLScore), polarity, subjectivity, no. of positive/negative words, word count, average word length, average words' length per sentence. The last category of features refers to N-grams and it is not included in Table IV. To compute the lexical features, we applied the following steps:

- Lowercasing;
- Remove punctuation marks and numbers;
- Tokenization;
- Lemmatisation and Stemming;
- Removing stop words and sparse words;
- Extracting N-grams using TfidfVectorizer provided by scikit-learn feature extraction package [27].

$Fmeasure$ [21], RIX, LIX indexes [22] and $CLScore$ [23] were calculated by using the formulas described in Equation 1, Equation 2, and Equation 3 respectively.

TABLE IV
CHARACTERISTICS

1	Word count (headline);
2	Average words' length (headline);
3	Finding punctuation patterns ("!?", "?!", "...", "***", "!!!", "???", "(", ")", "$") (headline);
4	No. of common nouns (headline and "bait" message);
5	No. of proper nouns (headline and "bait" message);
6	No. of common nouns with the syntactical tag of subject (headline and "bait" message);
7	No. of proper nouns with the syntactical tag of subject (headline and "bait" message);
8	No. and presence (boolean value) of question marks found in (headline and "bait" message);
9	No. and presence (boolean value) of exclamation marks found in (headline and "bait" message);
10	No. of uppercases (headline and "bait" message);
11	*Fmeasure* (headline);
12	*LIX* and *RIX* indexes (headline and "bait" message);
13	*CLScore* ("bait" message);
14	Presence (boolean value) of demonstratives (headline);
15	Presence (boolean value) of personal pronouns (headline);
16	If title starts with an adverb (boolean value) or with a numeral (boolean value) (headline);
17	No. of acronyms (headline and "bait" message);
18	Average words' length per statement ("bait" message);
19	No. of numerals (headline);
20	Polarity and subjectivity (headline, "bait" message and article's content);
21	No. of negative, no. of positive words (headline and "bait" message).

$$fmeasure = \frac{nounFreq + adjectiveFreq + prepositionFreq + articlesFreq}{2} + \frac{-(pronounsFreq + verbsFreq + adversFreq + injectionsFreq) + 100}{2} \quad (1)$$

$$RIX = \frac{LW}{S}; LIX = \frac{W}{S} + \frac{(100 * LW)}{W}, \quad (2)$$

where W is the total number of words, S is the number of sentences, and LW is the number of long words (long words are considered words with more than 7 characters).

$$CLScore = 5.88 \cdot L - 29.6 \cdot S - 15.8, \quad (3)$$

where L is the average number of letters per 100 words and S is the average number of sentences per 100 words. The coefficients were set according to the authors statistical experiments and research in order to work with English language [23].

Normalisation. The normalization process occurs in two stages: relabelling some of the text measures (LIX and RIX indexes) as the literature recommends it [6], normalizing all data to fit into the [0,1] interval. Analyzing our data, we observed that most features are natural numbers, number of a specific part of speech, or a syntactical tag. These numbers do not differ too much between the records, they do not have out-liners and they have finite values. Some other features are Boolean values that are easily mapped to 0.0 for the False value and to 1.0 for True. LIX index computes values for a five-point readability scale as presented in Table V. These values are then mapped to natural numbers between 0 and 4. RIX index is considered more accurate, because its values are split into thirteen formality levels. These values are mapped to a set of intervals determined by: 0.2, 0.5, 0.8, 1.3, 1.8, 2.4, 3.0, 3.7, 4.5, 5.3, 6.2, 7.2; as seen in Table VI.

TABLE V
LIX INDEX MAPPING VALUES

Readability level	Value	Mapped to
very easy	0-24	0
easy	25-34	1
standard	35-44	2
difficult	45-54	3
very difficult	>55	4

TABLE VI
RIX INDEX MAPPING VALUES

Value interval	Mapped to
<0.2	0
0.2-0.5	1
0.5-0.8	2
0.8-1.3	3
1.3-1.8	4
1.8-2.4	5
2.4-3.0	6
3.0-3.7	7
3.7-4.5	8
4.5-5.3	9
5.3-6.2	10
6.2-7.2	11
>7.2	12

Finally, all data was normalized using a MinMaxScaler, a normalization tool provided by Python's sklearn.preprocessing package before feeding the data into the intelligent classifier and training it. The MinMaxScaler is normalizing the data considering the mathematical formula presented in Equation 4.

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)} \quad (4)$$

where x represents a variable vector, a single feature in the intelligent model.

Intelligent Models. The related work analysis reflects that the most frequently applied intelligent classification algorithms are: Convolutional Neural Networks [18], LSTM (Long Short Term Memory), biGRU networks [14], Gradient Boosted Decision Trees [6], Random Forest, Logistic Regression, Naive Bayes [9] and Support Vector Machine. Considering this analysis, we chose to use Naive Bayes [9], Logistic Regression [9], [17], [31], Decision Tree [6], [7], Random Forest [9], [31], [7], [15], [17] and Support Vector Machine [15], [7].

Random Forest is an intelligent algorithm, made of multiple decision trees, each of them being built based on a set of features. The mechanism is called feature bagging and it is trying to avoid a high correlation between the trees [10]. Random Forest aims to have a low bias, high variance and tries to solve overfitting through their voting system [32]. In the present approach, we chose an already implemented algorithm from Python's library, sckit-learn [27].

When adjusting parameters for the RandomForestClassifier, we run multiple tests and we select them as it can be seen below:

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
   criterion='entropy', max_depth=None, max_features='auto',
   max_leaf_nodes=None, min_impurity_decrease=0.0,
   min_impurity_split=None, min_samples_leaf=1,
   min_samples_split=2, min_weight_fraction_leaf=0.0,
   n_estimators=150, n_jobs=-1, oob_score=True, random_state=0,
   verbose=0, warm_start=False)
```

Table VII contains all parameters used for calibrating the Random Forest Classifier, their values, a brief description, and motivation for the value chosen. The final values were set according to the official documentation [27], [33] and our empirical observations on the test results.

Logistic regression is one of the simplest and easiest algorithms to be applied for a classification problem. It is based on the Linear Regression and on the sigmoid function to better manage outlier values [34]. The formula for Logistic Regression is represented in Equation 5. In our approach, we modelled Logistic Regression using Python's scikit-learn [27] linear-model package.

$$p(t) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \quad (5)$$

Naive Bayes is a probabilistic algorithm often used for classification problems, such as: Spam Mail, Sentiment Analysis etc. It is based on the Bayes Theorem, where $X = (x_0, x_1, x_2, ..., x_n)$ is a vector representing the input features and Y is the output label. We want to use X to predict Y and we start by assuming that X consists of just independent features [35]. Naive Bayes Classifier is based on the equation 6. Because there are multiple Naive Bayes distributions, we chose the Gaussian distribution and we used GaussianNB already implemented in Python [27].

TABLE VII
PARAMETERS CALIBRATION

Parameter	Value	Observations
bootstrap	True (default)	Will split the input samples between trees in the constructing process
class_weight	'balanced'	The results proportion are inversely proportional to the class frequency
criterion	'entropy'	The tree construction will be done using information gain
max_depth	None (default)	Allows trees to not have a maximum depth, meaning the leaves will have a single output class
max_features	'auto' (default)	Allows the algorithm to automatically adjust the maximum numbers of features used in a split
max_leaf_nodes	None (default)	Lower impurity rate when the tree growth process takes place
min_impurity_decrease	0.0 (default)	Represents the minimum degree for impurity decrease when making a split in the tree
min_impurity_split	None	There will not be a threshold value to stop the trees' growth
min_samples_leaf	1 (default)	Represents the dimension of the terminal nodes of the tree
min_samples_split	2 (default)	Implies having at least 2 samples to make a split in the tree
min_weight_fraction_leaf	0.0 (default)	The same signification as "min_samples_leaf", but represents a percent
n_estimators	150	The value was chosen based on several test results
n_jobs	-1	Refers to the number of CPU cores used in the learning process. Value -1 means that all available CPU cores will be used
oob_score	True	The algorithm will use cross validation during training
random_state	0	Implies using a random instance or a given one in the bootstrap process
verbose	0 (default)	Refers to the printed output given when running the classifier
warm_start	False (default)	Set to False means creating new estimators at each run and not adding estimators to a previous instance of the classifier. We set this property to False such that the tests will give us proper results, not influencing one another

$$P(Y \vee X) = \frac{P(Y)\prod_i P(X_i \vee Y)}{P(X)} \tag{6}$$

Decision Trees are acyclic graphs, undirectly connected. They have a node called the root and the final nodes, which do not have any children are annotated as leaves. These structures are often used for supervised classification problems where the input is categorical data. Decision trees are flexible when training with incomplete and noisy data, achieving good results. They can also be used with continuous input, in this case the decision nodes will be labeled with an interval of values [36]. The present approach based its implementation on the Sklearn package [27] and the parameters we used to configure the decision tree classifier can be seen below. The criterion entropy will use information gain to compute the quality of a tree split.

```
DecisionTreeClassifier(random_state=0, criterion="entropy")
```

Support Vector Machine (SVM) is often used for big data analysis, especially in text analysis. Support Vector Machines are linear classifiers that identify the separation hyperplane for the negative and positive class [37]. SVM wants to find a linear function $f(x) = w \cdot x + b$, where w is the weight vector, such that:

$$y_i = \begin{cases} 1; & \text{if } w \cdot x_i + b \geq 0 \\ -1; & \text{if } w \cdot x_i + b < 0 \end{cases} \tag{7}$$

In the present approach, we used Support Vector Classifier implemented in the sklearn library with a linear kernel. The kernel function is the similarity function, which computes the distance between two points.

IV. RESULTS AND COMPARATIVE ANALYSIS

Metrics. To evaluate the performance of our intelligent models, we consider the following metrics, which are the same metrics as those used in The Clickbait Challenge evaluation: accuracy, precision, recall, F1 score, mean squared error (MSE) [38] and normalized mean squared error (NMSE) [39]. The results obtained with our algorithms are presented in Table VIII. These results were calculated as the average of 10 different program executions.

Analysis. Taking into consideration The Clickbait Challenge results [24] we would be classified: on the second place according to recall an F1 score; on the third place based on precision. Considering MSE, we will be the last clickbait detection approach. We need to mention that the metrics computed by The Clickbait Challenge are in relation to their testing set, which is not available to the public. Moreover, the task addressed during the Clickbait Challenge was just clickbait detection, without considering any multilanguage related aspects. We compute our results on a balanced testing set composed of 593 clickbait samples and 593 non-clickbait samples.

Even though the datasets used are different and they do not take into consideration a cross-language environment, we can compare our approach with [6], which scores a 74.9% accuracy (75% precision, 76% recall, and 74.9% F1 score) using a special type of Decision Tree (Gradient Boosted Decision Trees). Our simple decision tree scores about 62-63% accuracy (60-63% precision, 60-73% recall, and 60-61% F1 score). The database was labelled by volunteers and it contains news from: The Post, The New York Times, CBS, Forbes, The Huffington etc., having in total 1349 clickbaits and 2724 non-clickbaits.

One of the best clickbait detection solutions is described in [7], having 93% accuracy (95% precision, 90% recall, 93% F1 score) for a support vector machine. Even though they score one of the best results in the clickbait literature, their model does not consider language-independent features, aspect on which we focused the most. They obtained the best accuracy for the model using all features, in contrast we obtained the best accuracy for lexical ones. Still, our best algorithm is also Support Vector Machine. For the Random Forest, they obtained (for all features) 92% accuracy (94% precision, 91% recall, and 89% F1 score), compared to our

TABLE VIII
RESULTS OBTAINED

Grammatical + Text Measures + Orthographic						Lexical						Grammatical + Text Measures + Orthographic + Lexical					
	RF	**LR**	**NB**	**DT**	**SVM**		**RF**	**LR**	**NB**	**DT**	**SVM**		**RF**	**LR**	**NB**	**DT**	**SVM**
Accuracy	70.15%	**71.3%**	67.51%	62.09%	70.57%	**Accuracy**	70.06%	**73.93%**	67.27%	63.15%	**73.93%**	**Accuracy**	**71.81%**	68.15%	63.18%	62.12%	68.06%
Recall	67.19%	**71.41%**	73.88%	63.54%	71.14%	**Recall**	79.25%	**71.08%**	61.44%	73.18%	**71.08%**	**Recall**	**68.45%**	73.28%	59.48%	60.09%	77.38%
F1 score	69.67%	**71.75%**	69.4%	63%	70.1%	**F1 score**	72.58%	**73.29%**	65.38%	66.51%	**73.29%**	**F1 score**	**70.99%**	70.09%	62.01%	60.39%	70.82%
Precision	72.7%	**72.41%**	65.65%	62.87%	69.38%	**Precision**	66.95%	**75.64%**	69.86%	60.95%	**75.64%**	**Precision**	**74.2%**	67.37%	65.09%	61.09%	62.53%
MSE	0.29	**0.28**	0.32	0.37	0.29	**MSE**	0.29	**0.26**	0.32	0.36	**0.26**	**MSE**	**0.28**	0.31	0.36	0.37	0.31
NMSE	1.24	**1.11**	1.16	1.45	1.22	**NMSE**	1.01	**1.09**	1.47	1.22	**1.09**	**NMSE**	**1.2**	1.12	1.58	1.66	1.07

approach, in which we obtained (for all features) 71.81% accuracy (74.2% precision, 68.48% recall, 70.99% F1 score). In terms of Decision Tree, we obtained the best accuracy for lexical features: 63.15% accuracy (60.95% precision, 73.18% recall, 66.51% F1 score). Their study has 90% accuracy (91% precision, 89% recall, 90% F1 score) for Decision Tree algorithm. Because our model has fewer features, it is inferior in terms of accuracy metrics when compared to other literature solutions. However, it relies only on features we considered to be language independent. Moreover, the present model uses just TF-IDFs and does not apply any other methods to automatically extract relevant features from text such as Word2Vec, GloVe or the BERT model, which are known to be significantly more accurate. When building the dataset, the authors used English Wikinews articles for the non-clickbait records and for the clickbait records they put volunteers to annotate a selection of news from multiple media outlets (ViralStories, BuzzFeed, Upworthy, ViralNova and Scoopwhoop). It is worthy to mention that paper [7] used some language-dependent text characteristics, such as: Internet slang, Common bait phases, Hyperbolic terms; and a specialized dataset - in contrast to our approach that selected intelligent model's features having a multilanguage context in mind.

The solution proposed in [9] obtained an ROC-AUC of 0.79 (precision 76%, recall 76%) for the top 1000 features used with a Random Forest algorithm. This solution was not developed to take into consideration a multilingual environment, compared to our approach which was constructed starting from a multilingual point of view. Our Random Forest algorithm performs best for all features: precision 74.2%, recall 68.45%. The aforementioned paper also tested other algorithms such as: Logistic Regression and Naive Bayes. The best result obtained for Logistic Regression is 0.73 ROC-AUC (precision 71% and recall 65%), while our model performs best for 75.64% precision and 71% recall (lexical features). Our best solution for Naive Bayes is 65.65% precision and 73.88% recall resulted for grammatical, orthographic and text measure features. Naive Bayes in paper [9] runs best for the top 100 characteristics with a precision score of 72% and 65% recall (ROC-AUC 0.72). All tests were run on a dataset containing 2992 tweets manually labelled by three different volunteers. Tweets were extracted from well-known journalistic publications, such as: BBC News, New York Times, ABC News, CNN, Fox News, and so on.

In table IX, we present the most relevant features as determined by the RFC (Random Forest Classifier) together with their computed importance. Albeit initially there were taken into consideration over 52 features, in the end we relied only on those features that have their importance greater than 1%. As presented in table IX, the most important feature is the count of proper nouns extracted from the postText attribute of the tweet. This has been already mentioned previously in the clickbait scientific literature, [7] also stated that the use of a small number of proper nouns is a clickbait characteristic. A clickbait related article is scarce in detail, omitting intentionally relevant information about the events, persons, or place the article is talking about, the presented information being in general confusing and vague. In contrast, high-quality articles use a greater number of proper nouns just to assure that the events are accurately presented. Moreover, the sixth feature is also related to the number of proper nouns within the title of the article.

The second and fourth features considering their importance are $CLScore$ and $Fmeasure$. These are extracted from the postText and the headline, respectively, both of them measuring the article's informality. A previous study [6] also placed them in the top 10 features suitable for clickbait detection.

TABLE IX
TOP 25 FEATURES' IMPORTANCE
FOR THE RANDOM FOREST CLASSIFIER

No.	**Feature**	**Importance percent (%)**
1	No. of proper nouns in postText	7.769
2	*Fmeasure* headline	6.569
3	Subjectivity content	6.307
4	*ClScore* posttext	6.178
5	Polarity content	5.845
6	No of proper nouns in headline	5.48
7	Average words number in headline	5.47
8	Average no. words per statement in posttext	4.014
9	No. words in headline	3.883
10	Subjectivity posttext	3.77
11	Polarity headline	3.583
12	No. common nouns in posttext	3.335
13	Subjectivity headline	3.218
14	Polarity posttext	3.206
15	*RIX posttext*	3.177
16	No. uppercase words in posttext	2.81
17	No of common nouns in headline	2.804
18	No. acronyms in posttext	2.793
19	*RIX headline*	2.606
20	*LIX posttext*	2.186
21	No. of uppercase words in headline	2.082
22	No. of acronyms in headline	1.907
23	*LIX headline*	1.74
24	No. numbers headline	1.15
25	No. positive words in posttext	1.057

TABLE X
TOP 15 FEATURE RANKING FOR EACH ALGORITHM

Feature	Ranking				
	RF	LR	DT	NB	SVM
No. proper nouns in posttext	**1**		**1**	**14**	
Fmeasure headline	**2**		**5**	**1**	
Subjectivity content	**3**	**1**	**2**	**7**	**1**
Clscore posttext	**4**		**3**	**5**	
Polarity content	**5**	**5**	**4**	**11**	**6**
No of proper nouns in headline	6				
Average words number in headline	7		7		
Average no. words per statement in posttext	**8**	**10**	**6**	**9**	
No. words in headline	9		8		
Subjectivity posttext	**10**	**3**	**9**		**3**
Polarity headline	**11**		**10**	**2**	**14**
No. common nouns in posttext	**12**		**11**	**13**	
Subjectivity headline	**13**	**13**	**13**	**12**	**15**
Polarity posttext	**14**	**15**	**14**	**3**	
RIX posttext	**15**			**6**	**11**
No of ? in headline		**2**	**12**		**2**
Presence of ? in posttest		4			8
Headline starts with adverb		6			7
Headline starts with number		7			9
No. positive words in posttext		8			
No. of ? in posttext		9			13
No of common nouns in headline		11			5
No. negative words in headline		12			12
Presence of personal pronouns in headline		14		15	
RIX headline			15	4	
LIX headline				8	10
LIX posttext				10	
No. positive words in posttext					4

The high importance of these features is related to the use of informal language, vulgar, and slang words. To increase the reader's curiosity and to draw more attention to the bait, sometimes publishers overuse an informal or colloquial language. In contrast, non-clickbait articles use a more formal, rigorous language, maybe even specialized, to objectively report a series of events.

Third and fifth features are represented by subjectivity content and, respectively, by polarity content. Subjectivity and polarity are important features in clickbait detection because they measure emotions or how much a news article makes use of emotions. Traditional news is usually objective, there are no sentiments transmitted when narrating facts. However, clickbait and low-quality media articles do use emotion to engage with the readers, trying to monopolize their views and opinions. Article [40] states that articles not related to news are more shared if they are transmitting a positive sentiment and articles containing news-related topics are shared more if the predominant sentiment is negative. Either way, emotion arousal is an important tool to attract online attention and readers' clicks.

The seventh feature in our hierarchy considering its importance is the average word length within the title. Previous studies have also identified the relevance of this characteristic in clickbait detection. Paper [7] reveals that in high-quality news the average word length is 10, while in clickbait headlines its value is approximately 7. This is a normal consequence of the fact that, in the case of clickbait articles, the title sometimes contains simplified words such as abbreviations, acronyms, or slang. An important observation is that the most relevant characteristics, the number of proper nouns (postText) and the CLScore (postText), were extracted from the information that determines the user's engagement (i.e. the bait link itself). The second, sixth, and seventh ones as importance, Fmeasure (headline), number of proper nouns (headline), and average word length (headline), were calculated from the bait article's title which can be the same as the bait link text, slightly different, or completely rewritten using different words. Moreover, the content (the target paragraphs) is also important in respect to the way the article is written in terms of subjectivity (third feature) and polarity (fifth feature) or the sentiments transmitted.

Features presented in table IX were grouped together based on their importance: less important features (importance under 2%), features with medium importance (between 2% and 5%), and the most relevant features having their importance greater than 5%. Although a clear decision about classifying a link as clickbait or not can be taken based on the top most relevant features, to solve the overfitting problem, features having a relevance under 2% have also to be taken into consideration. The relevance of these features could also depend on the used dataset, even if they have a lower importance in our experiments, they could have a higher impact if another dataset is used.

V. CONCLUSIONS AND FUTURE WORK

Through this paper, we presented some new advances in clickbait and fake news detection using language-independent

strategies. Additionally, we demonstrated the viability of using such an approach. The obtained results (71-74% accuracy, 72-75% precision, 68-71% recall, and 70-73% F1 score) achieved using Logistic Regression, Support Vector, and Random Forest Classifier are supportive enough to encourage further investigation in this direction. Our feature importance hierarchy is confirmed by similar results obtained in the scientific literature. One of the benefits of our proposed approach is that it allows, by using only language-independent features, the training of an intelligent classifier on a dataset containing sample articles written in more than just one language. To determine the most relevant features, only universal characteristics were taken into consideration, a natural language processor being used to annotate the universal parts of speech tags.

Although the obtained accuracy and precision could be evaluated as being very good, to maximize them, future work has to be done. This implies testing the proposed methodology with other artificial intelligence algorithms such as: Neural Networks, K-Nearest Neighbour and experiments with different types and configuration of neural networks, such as: Recurrent Neural Networks, bi-GRU, LSTM (Long Short Term Memory model).

Because of our encouraging results, we want also to evaluate our proposed technique on a different data set that is currently being built and populated with clickbait samples provided by [1] and non-clickbait samples provided by the plugin described in [41], [42] and [43].

Finally, we intend to implement a browser plugin that will signal clickbaits found in online media, helping users to have a better, undeceived online experience.

ACKNOWLEDGMENT

This work was supported by the following research grant: "Upgrade of the Cloud Infrastructure of the Babeş-Bolyai University Cluj-Napoca in Order to Develop an Academic Management and Decision Support Integrated System Based on Big&Smart Data - SmartCloudDSS" - POC/398/1/1/124155 - a Project Co-financed by the European Regional Development Fund (ERDF) through the Competitiveness Operational Programme 2014-2020.

REFERENCES

[1] D. Bufnea and D. Şotropa, "A community driven approach for click bait reporting," in *26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split – Supetar (Island of Brac), Croatia, September 2018. doi: 10.23919/SOFTCOM.2018.8555759 pp. 1–6.

[2] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *SIGKDD Explor. Newsl.*, vol. 19, no. 1, pp. 22–36, September 2017. doi: 10.1145/3137597.3137600

[3] D. Gilbert, D. Krull, and P. Malone, "Unbelieving the unbelievable: Some problems in the rejection of false information," *Journal of Personality and Social Psychology*, vol. 59, pp. 601–613, 10 1990. doi: 10.1037/0022-3514.59.4.601

[4] C. I. Coste, D. Bufnea, and V. Niculescu, "A new language independent strategy for clickbait detection," in *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Hvar, Croatia (online), 2020. doi: 10.23919/SoftCOM50211.2020.9238342 pp. 1–6.

[5] C. I. Coste, "Controlling the click bait," in *Proceedings of the International Student Conference StudMath-IT*, Arad, Romania, November 2018, pp. 11–17.

[6] P. Biyani, K. Tsioutsiouliklis, and J. Blackmer, ""8 amazing secrets for getting more clicks": Detecting clickbaits in news streams using article informality," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI'16. AAAI Press, 2016, pp. 94–100.

[7] A. Chakraborty, B. Paranjape, S. Kakarla, and N. Ganguly, "Stop clickbait: Detecting and preventing clickbaits in online news media," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2016. doi: 10.1109/ASONAM.2016.7752207 pp. 9–16.

[8] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, Maryland: Association for Computational Linguistics, June 2014. doi: 10.3115/v1/P14-5010 pp. 55–60. [Online]. Available: https://www.aclweb.org/anthology/P14-5010

[9] M. Potthast, S. Köpsel, B. Stein, and M. Hagen, "Clickbait detection," in *Ferro N. et al. (eds) Advances in Information Retrieval. ECIR 2016. Lecture Notes in Computer Science*, vol. 9626. Springer, Cham, 03 2016. doi: 10.1007/978-3-319-30671-1_72. ISBN 978-3-319-30670-4 pp. 810–817.

[10] D. Denisko and M. M. Hoffman, "Classification and interaction in random forests," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 115, no. 8, pp. 1690–1692, Feb 2018. doi: 10.1073/pnas.1800256115. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/29440440

[11] M. Potthast, T. Gollub, M. Hagen, and B. Stein, "The clickbait challenge 2017: Towards a regression model for clickbait strength." s.l. : CoRR, 2018, Vol. abs/1812.10847., 2018.

[12] A. Omidvar, H. Jiang, and A. An, "Using neural network for identifying clickbaits in online news media," in *Information Management and Big Data, 5th International Conference, SIMBig 2018, Lima, Peru, September 3-5, 2018, Proceedings*, ser. Communications in Computer and Information Science, vol. 898. Springer, 2018. doi: 10.1007/978-3-030-11680-4_22 pp. 220–232.

[13] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*, 2020, ch. Bidirectional recurrent neural networks. [Online]. Available: https://d2l.ai

[14] Y. Zhou, "Clickbait detection in tweets using self-attentive network," *arXiv preprint arXiv:1710.05364*, 2017.

[15] A. Grigorev, "Identifying clickbait posts on social media with an ensemble of linear models," *CoRR*, vol. abs/1710.00399, 2017. [Online]. Available: http://arxiv.org/abs/1710.00399

[16] C. I. Coste, "Online bad habits: fake news and clickbait," in *Proceedings of the Student Interdisciplinary Conference The European Union and Global Order*, Cluj-Napoca, Romania, April 2019.

[17] V. Indurthi, B. Syed, M. Gupta, and V. Varma, "Predicting clickbait strength in online social media," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 4835–4846.

[18] H.-T. Zheng, J.-Y. Chen, X. Yao, A. K. Sangaiah, Y. Jiang, and C.-Z. Zhao, "Clickbait convolutional neural network," *Symmetry*, vol. 10, no. 5, p. 138, 2018.

[19] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[20] A. Anand, T. Chakraborty, and N. Park, "We used neural networks to detect clickbaits: You won't believe what happened next!" in *Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2017, Proceedings"*, ser. LNCS. Springer Verlag, 2017. doi: 10.1007/978-3-319-56608-5_46. ISBN 9783319566078 pp. 541–547.

[21] F. Heylighen and J.-M. Dewaele, "Formality of language: definition, measurement and behavioral determinants," Center "Leo Apostel", Free University of Brussels, Tech. Rep., 1999.

[22] J. Anderson, "Lix and rix: Variations on a little-known readability index," *Journal of Reading*, vol. 26, no. 6, pp. 490–496, 1983. [Online]. Available: http://www.jstor.org/stable/40031755

[23] M. Coleman and T. L. Liau, "A computer readability formula designed for machine scoring." *Journal of Applied Psychology*, vol. 60, no. 2, pp. 283–284, 1975. doi: 10.1037/h0076540. [Online]. Available: https://doi.org/10.1037/h0076540

[24] Bauhaus-Universität Weimar, *The Clickbait Challenge*, 2017 (visited on May 5, 2020). [Online]. Available: https://www.clickbait-challenge.org

[25] P. Qi, T. Dozat, Y. Zhang, and C. D. Manning, "Universal Dependency parsing from scratch," in *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Brussels, Belgium: Association for Computational Linguistics, Oct

2018. doi: 10.18653/v1/K18-2016 pp. 160–170. [Online]. Available: https://www.aclweb.org/anthology/K18-2016
[26] S. Loria, *TextBlob: Simplified Text Processing*, 2020 (visited on November 30, 2020). [Online]. Available: https://textblob.readthedocs.io/en/dev/
[27] *scikit-learn Machine Learning in Python*, 2017, (visited on December 9, 2020). [Online]. Available: https://scikit-learn.org/stable/
[28] E. K. Steven Bird, Edward Loper, *Natural Language Processing with Python*, 1st ed. O'Reilly Media, Inc., 2009. ISBN 0596516495. [Online]. Available: https://www.nltk.org/
[29] *StanfordNLP - Python NLP Library for Many Human Languages*, 2017, (visited on May 12, 2019). [Online]. Available: https://stanfordnlp.github.io/stanfordnlp/
[30] Universal Dependencies, *Universal POS tags*, 2014 (visited on December 16, 2020). [Online]. Available: https://universaldependencies.org/u/pos/
[31] X. Cao, T. Le, and J. Zhang, "Machine learning based detection of clickbait posts in social media," *CoRR*, vol. abs/1710.01977, 2017. [Online]. Available: http://arxiv.org/abs/1710.01977
[32] G. Louppe, "Understanding random forests: From theory to practice," Ph.D. dissertation, University of Liège, Faculty of Applied Sciences, 2015. [Online]. Available: https://arxiv.org/abs/1407.7502
[33] T. Plapinger, *Tuning a Random Forest Classifier*, 2017 (visited on November 28, 2020). [Online]. Available: https://medium.com/@taplapinger/tuning-a-random-forest-classifier-1b252d1dde92
[34] S. Menard, *Logistic regression: From introductory to advanced concepts and applications*. Sage, 2010.
[35] G. I. Webb, E. Keogh, and R. Miikkulainen, "Naïve bayes." *Encyclopedia of machine learning*, vol. 15, pp. 713–714, 2010.
[36] Y.-Y. Song and L. Ying, "Decision tree methods: applications for classification and prediction," *Shanghai archives of psychiatry*, vol. 27, no. 2, p. 130, 2015.
[37] W. S. Noble, "What is a support vector machine?" *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.
[38] A. Mishra, *Metrics to Evaluate your Machine Learning Algorithm*, 2018 (visited on November 23, 2020). [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234
[39] *NMSE*, (visited on May 14, 2020). [Online]. Available: https://rem.jrc.ec.europa.eu/RemWeb/atmes2/20b.htm
[40] G. Lockwood, "Academic clickbait: Articles with positively-framed titles, interesting phrasing, and no wordplay get more attention online," *The Winnower*, vol. 3, 2016.
[41] I. Bădărînză, A. Sterca, and D. Bufnea, "A dataset for evaluating query suggestion algorithms in information retrieval," in *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, 2019. doi: 10.23919/SOFTCOM.2019.8903906 pp. 1–6.
[42] I. Bădărînză, A. Sterca, and F. M. Boian, "Using the user's recent browsing history for personalized query suggestions," in *2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split – Supetar (Island of Brac), Croatia, 2018. doi: 10.23919/SOFTCOM.2018.8555774 pp. 1–6.
[43] I. Badarinza, A. Sterca, and F. Boian, "The role of the user's browsing and query history for improving mpc-generated query suggestions," *Journal of Communications Software and Systems*, vol. 15, no. 1, pp. 26–33, March 2019. doi: 10.24138/jcomss.v15i1.608

Claudia Ioana Coste is a Master Student in Distributed Systems in Internet, at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Cluj-Napoca, Romania. She holds a Bachelor's degree in Computer Science and graduated with 1st class honors. She has published three research articles on topics such as: clickbait and fake news detection and information retrieval. She is currently focusing on natural language processing, text mining, and data modeling. She also works as a full-stack developer at Tapptitude mobile application development agency.

Darius Bufnea is a lecturer at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Romania. He received the Ph.D. degree from Babeş-Bolyai University in 2008. He was the director of one research grant funded by the Romanian funding agency and a member of another five. He has authored over 35 research papers on networking, congestion control, information retrieval, parallel computing, and web technologies. He is the coauthor of four books that cover different computer science related topics. His current focus and interest is on cybersecurity.