IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 November 2023, accepted 10 January 2024, date of publication 12 January 2024, date of current version 9 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3353692

RESEARCH ARTICLE

Assessing Accuracy: A Study of Lexicon and Rule-Based Packages in R and Python for Sentiment Analysis

AMIN MAHMOUDI[1], DARIUSZ JEMIELNIAK[1,2], AND LEON CIECHANOWSKI[1,3]
[1]Management in Networked and Digital Societies (MINDS) Department, Kozminski University, 03-301 Warsaw, Poland
[2]Berkman-Klein Center for Internet and Society, Harvard University, Cambridge, MA 02138, USA
[3]MIT Center for Collective Intelligence, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Corresponding author: Amin Mahmoudi (amin_mahmoudy@yahoo.com)

This work was supported by Narodowe Centrum Nauki (National Science Centre, Poland) under Grant 2020/38/A/HS6/00066.

ABSTRACT Sentiment analysis has become a focal point of interdisciplinary research, prompting the use of diverse methodologies and the continual emergence of programming language packages. Notably, Python and R have introduced comprehensive packages in this realm. In this study, we analyze established packages in these languages, focusing on accuracy while also considering time complexity. Across experiments conducted on seven distinct datasets, a crucial revelation surfaces: the accuracy of these packages significantly varies depending on the dataset used. Among these, the 'sentimentr' package consistently performs well across diverse datasets. Generally, Python libraries showcase superior processing speed. However, it's essential to note that while these packages adeptly classify sentences as positive or negative, capturing sentiment intensity proves challenging. Our findings highlight a prevalent trend of overfitting, where these packages excel on familiar datasets but struggle when faced with unfamiliar ones.

INDEX TERMS Sentiment analysis, lexicon and rule based, sentiment analysis by R, sentiment analysis by python, VADER, sentimentr.

I. INTRODUCTION

A. BACKGROUND

Sentiment analysis stands as a fundamental tool in computational social science and social networks analysis [1]. Its applications span predicting stock movements, gauging public sentiment on vaccine hesitancy, environmental concerns, and cultural conflicts [2], [3], [4], [5], [6], [7]. The accuracy of sentiment measurement significantly impacts hypotheses and theories across various fields like psychology, sociology, data science, and marketing [8]. Consequently, standardized measures and tools for sentiment analysis are crucial.

Sentiment analysis predominantly employs two primary approaches: machine learning and lexicon-based methods [9], [10]. Machine learning methods, incorporating supervised learning techniques and pre-training methodologies, offer advantages by eliminating the need for exhaustive lexicon construction. However, they require well-validated training data, demanding significant effort and resources, while also operating as a black box, making their operational mechanics less intuitive.

On the other hand, lexicon-based approaches forego training data but face challenges in constructing and validating lexicons [11]. While avoiding the complexities of training, they heavily rely on the accuracy and comprehensiveness of their underlying lexicons, introducing potential biases and inaccuracies from these sources.

B. MOTIVATION

Researchers continuously strive to enhance the precision of sentiment analysis methodologies, leading to the development of numerous packages in two central programming languages for data science: Python and R. This study specifically focuses on evaluating lexicon and rule-based approaches within Python and R, attracted by their inherent simplicity. Although extensive research compares the

The associate editor coordinating the review of this manuscript and approving it for publication was Camelia Delcea.

accuracy of available packages in this field, a challenge lies in comparing packages across these distinct programming languages. Additionally, the absence of a universally established ground truth for sentiment intensity complicates the search for studies examining package accuracy in this regard.

The assessment of time complexity represents another significant challenge in sentiment analysis, one that remains underexplored in existing studies. The primary goal of this study is to conduct a comprehensive analysis of well-established and widely used sentiment analysis packages within the R programming language, such as 'sentimentr', 'SentimentAnalysis', 'syuzhet', 'qdap', 'meanr', and 'Rsentiment'. Similarly, we scrutinize notable packages in the Python programming language, including 'vadersentiment', 'pattern', 'flair', 'NLTK', 'TextBlob', 'Stanza', and 'Affin'. With a range of inherent advantages and drawbacks in lexicon-based approaches, an empirical evaluation of their accuracy becomes crucial. As a result, these methods undergo evaluation across seven distinct datasets, fostering a comprehensive and holistic assessment.

C. CONTRIBUTIONS

The paper's principal contributions are outlined as follows:

- **Cross-Linguistic Package Comparison**: The paper undertakes a comprehensive comparison of sentiment analysis packages in two distinct programming languages, R and Python.
- **Novel Exploration of Sentiment Intensity**: In addition to evaluating polarity accuracy, this study pioneers the analysis of the precision of existing packages concerning sentiment intensity. This facet assumes significance in various domains, where discerning shifts in sentiment intensity over time holds value.
- **Unveiling the Overfitting Phenomenon**: This study uniquely addresses the overfitting quandary within lexicon and rule-based methodologies, shedding light on a previously unexplored aspect.
- **Time Complexity Evaluation**: An exploration of the time complexity associated with the aforementioned packages constitutes an integral component of this study's contributions.

D. PAPER STRUCTURE

The structure of this paper follows this sequence: the subsequent section reviews related works in this field, followed by a description of the methods employed in this study in Section III. Section IV elaborates on the results, while Section V provides a discussion about these results. In Section VI, we propose several research directions. The paper concludes with Section VII, encapsulating the key conclusions.

II. RELATED WORKS

Considering the paramount significance of sentiment analysis method and package accuracy, several studies have been dedicated to comparing these methods and packages against one another. In the work by [12], VADER lexicons were introduced and their accuracy was benchmarked against well-known lexicons including Hu Liu04, SCN, GI, SWN, LIWC, ANEW, and WSD. The results of their study unveiled VADER's prowess, particularly in social media texts, as it outperformed other lexicons. In a separate experiment, VADER was pitted against Naïve Bayes, Maximum Entropy, and SVM. The outcome of this comparison revealed VADER's superiority across four diverse datasets encompassing tweets, movie reviews, product reviews, and opinion news articles.

Similarly, in the study conducted by [10], a comparison was drawn among three classification algorithms: SVM, Stochastic Gradient Descent, and logistic regression, alongside two lexicons, AFFIN and VADER. The experiment focused on Trip Advisor hotel reviews, the results showcased SVM (TF-IDF) emerging as the frontrunner, boasting an impressive accuracy of 96%. Within the realm of unsupervised methods, VADER exhibited its supremacy by surpassing AFFIN with an accuracy of 88%.

In their work, [13] conducted a valuable study delving into sentiment analysis packages within the R environment. The investigation meticulously evaluated the functionality of five renowned packages: 'syuzhet', 'RSentiment', 'SentimentAnalysis', 'meanr', and 'sentimentr'. Notably, their comparison unearthed that 'sentimentr' employs a more sophisticated method to ascertain polarity scores, which, despite its intricacy, yields enhanced efficiency. Of particular significance, the experimental outcomes underscored the supremacy of 'sentimentr', showcasing its superior accuracy relative to other packages.

Likewise, Naldi in [14] undertook an exploration encompassing four packages: 'syuzhet', 'RSentiment', 'SentimentR', and 'SentimentAnalysis'. A common thread between these packages was identified—they all employ the bag-of-words methodology, wherein sentiment evaluation hinges on individual words within the text, dismissing considerations of syntax and grammar. Naldi's findings accentuated the preeminence of the 'sentimentr' package, while other packages grappled with accounting for negators. 'sentimentr', grounded in a lexicon-based approach, has garnered considerable research attention, culminating in numerous studies.

The lexicon-based approach was described by [15], emphasizing the process of establishing document orientation through the semantic orientation calculation of words or phrases within the document. They underscored the pivotal role of lexical items in conveying vital semantics, highlighting the significance of going beyond adjectives alone.

In a distinct study, Al-Shabi in [16] engaged in a comparison of five prominent lexicons through the analysis of Twitter data. The lexicons under scrutiny included SentiWordNet, VADER, SentiStrength, AFINN-111, and Liu and Hu. The experimental outcomes, as gauged against Stanford and Sandars datasets, unequivocally favored VADER, demonstrating its superior accuracy in contrast to

the other lexicons. Nonetheless, it's notable that a prevailing theme among many studies casts a shadow of doubt on the efficacy of dictionary-based methodologies. Young and Soroka in [17] discovered minimal concurrence and weak correlation between sentiment measurement dictionaries and expert annotations. Similarly, [18] reported that despite the widespread availability of sentiment dictionaries and readily employable tools, sentiment measurement derived from dictionaries exhibits diminished validity and accuracy when juxtaposed with machine learning and manual/crowd coding approaches.

In a different vein, Nguyen et al. in [11] executed a comparison between sentiment analysis outcomes derived from three machine learning techniques and three lexicon-based techniques. Their investigation uncovered lower accuracy for the lexicon-based approach. Likewise, [10] embarked on a comparative study between machine learning-based methods (e.g., SVM and logistic regression) and lexicon-based approaches (e.g., VADER and AFINN). Their research underscored the diminished accuracy of lexicon-based methods. This collection of findings has spurred heightened interest in machine learning techniques, particularly transformers like BERT, GPT3, etc., which have exhibited enhanced accuracy in sentiment analysis [19].

III. METHODS

A. DATA

The collected datasets for this study are categorized into two groups: datasets aimed at measuring accuracy in sentiment intensity and datasets utilized to assess the accuracy of polarity classification.

For the assessment of sentiment intensity accuracy, we acquired two datasets from [12]: the first being the Amazon dataset and the second, the New York editorial dataset. Both datasets furnish ground truth information for sentiment intensity.

Furthermore, we incorporated three distinct review datasets sourced from Amazon [20]. These datasets pertain to reviews across three diverse product categories: luxury products, software, and industrial and scientific items. The datasets encompass a range of information fields, including reviewer ID, product ID, reviewer name, vote, review text, product rating, review summary, and the time of the review (in UNIX time format). Within the context of this study, our primary focus lies on two specific information fields: the review text and the summary text. While these datasets do not explicitly provide ground truth sentiment intensity values, they do provide a summary field that encapsulates the essence of the text review. It is crucial to emphasize that the orientation of sentiment in the summary and text review should align. While exact intensity values might differ, a correlation is expected, assuming the packages and libraries function effectively. Consequently, this summary field is regarded as the gold standard. The utilization and the mechanism by which this field serves as the ground truth will be expounded upon in the evaluation methodology section.

For measuring polarity classification accuracy, we utilized two annotation-based datasets: movie reviews and Tweets concerning the first 2016 GOP Presidential Debate. These datasets contain ground truth labels indicating whether reviews convey positive or negative sentiments. It's important to note that these datasets do not provide sentiment scores; they are exclusively used to assess polarity classification accuracy.

It is important to note that certain packages faced difficulties in reading all review texts within some datasets. As a result, we considered a number of clean review texts in each dataset, enabling the application of sentiment computation functions from the packages. A statistical summary of these datasets is provided in Table 1.

TABLE 1. Datasets statistic.

Table	No of reviews
luxury products	11,763
Software products	4,330
industrial and scientific products	50,924
IMDB movie reviews	5,550
GOP Debate (GOP)	9,425
Amazon product review	1,830
New York editorials (NY)	3,708

B. SENTIMENT VALUE CALCULATION

In this study, we have chosen six prominent packages from R: 'sentimentr', 'SentimentAnalysis', 'syuzhet', 'qdap', 'meanr', and 'Rsentiment'. Additionally, we have selected seven libraries in Python: 'vadersentiment', 'pattern', 'flair', 'NLTK', 'TextBlob', 'stanza', and 'affin'. These packages and libraries are widely recognized and extensively utilized in the field of sentiment analysis. Notably, our selection criterion focused on packages and libraries that offer a continuous value to represent sentiment intensity. Each of these packages and libraries is equipped with a function designed for computing sentiment scores or for classifying polarity. In this section, we provide an overview of the functions employed for these purposes.

1) SENTIMENTR

Within the sentimentr package, the sentiment_by() function plays a pivotal role. Upon input of a text, this function yields several informative values pertaining to the text's characteristics. These values encompass element_id, word_count, standard deviation (sd), and ave_sentiment. Of notable significance, the ave_sentiment attribute of this function furnishes a continuous value representing the intensity of sentiment within the text.

2) SENTIMENTANALYSIS (SA)

As for the SentimentAnalysis (SA) package, the primary function for sentiment computation is analyzeSentiment(). When provided with a text, this function returns an array of values including WordCount, SentimentGI, NegativityGI,

PositivityGI, SentimentHE, NegativityHE, PositivityHE, SentimentLM, NegativityLM, PositivityLM, RatioUncertaintyLM, SentimentQDAP, NegativityQDAP, and PositivityQDAP. Notably, according to CRAN [21] and [22], this dictionary-based package leverages QDAP, Harvard IV, Loughran-McDonald, and Henry's Financial dictionaries. The values mentioned above correspond to these four aforementioned dictionaries. Within the scope of this study, our focus centers on the SentimentLM value derived from the Loughran-McDonald dictionary, which carries specific relevance to our analysis.

3) SYUZHET

The syuzhet package introduces two key functions: get_sentences(text) and get_sentiment (sentimentSC). The former takes a text as input and segments it into individual sentences. The latter, which receives the output from the first function (sentimentSC), generates a continuous sentiment intensity value for each sentence. To derive a comprehensive sentiment value for an entire paragraph, we calculate the mean sentiment value across all sentences within the paragraph.

4) QDAP

In the qdap package, a pivotal function named polarity takes center stage. When supplied with text, this function returns an array of values including all, total.sentences, total.words, ave.polarity, sd.polarity, and stan.mean.polarity. It's important to note that the ave.polarity attribute yields a sentiment value which remains continuous. This value has the potential to exceed the range of -1 to 1, thus allowing for values below -1 and above 1 to be returned.

5) MEANR

The main function of this package is score() which contains two parameters such as text and nthreads = meanr.nthreads(). The output includes four values such as positive, negative, score, and wc. The positive (negative) is the number of positive (negative) sentences and the score is the difference of these two values, as these values are integer, therefore this package does not provide a continuous value to show the sentiment intensity, it just classifies sentiment polarity.

6) RSENTIMENT

Within the Rsentiment package, the calculate_score() function plays a central role. However, the value it returns is not continuous; rather, it yields integer values that signify binary polarity. Specifically, a value of 0 denotes neutral sentiment, a positive value signifies positive sentiment, a negative value indicates negative sentiment, and 99 is indicative of sarcasm. This function categorizes sentences into six distinct categories: Positive, Negative, Very Positive, Very Negative, Sarcasm, and Neutral.

7) VADERSENTIMENT (VS)

For vaderSentiment, sentiment computation is facilitated through the SentimentIntensityAnalyzer function. This function, devoid of input, yields an analyzer that furnishes the polarity_scores() function. When provided with a text, this function returns four values: 'neg', 'neu', 'pos', and 'compound'. These values span the range of $[-1, 1]$, and the 'compound' value is particularly relevant for determining sentiment intensity.

8) TEXTBLOB

The TextBlob library utilizes the TextBlob() function, taking text as input and producing a blob object as output. This object can be employed to tokenize text into words and sentences. The sentiment intensity can be derived using the sentiment.polarity attribute of this object. The resulting values fall within the range of $[-1, 1]$.

9) NLTK

Similar to the vaderSentiment library, NLTK features the SentimentIntensityAnalyzer() function. This function returns an object for sentiment analysis. Subsequently, the polarity_scores() function within this object accepts a text and provides four values: 'neg', 'neu', 'pos', and 'compound'. These values are on a scale from $[-1$ to $1]$.

10) PATTERN

Pattern offers a primary function for sentiment analysis, namely sentiment (). This function takes a text as input and returns two continuous values. The first value indicates the sentiment intensity, while the second value reflects subjectivity. If the text contains personal opinions, the subjectivity value is higher compared to texts containing factual information. These values also fall within the range of $[-1$ to $1]$.

11) FLAIR

For sentiment analysis tasks, Flair offers a main function called Sentence (text). This function takes the input text and provides sentiment labels such as positive and negative, along with a sentiment value that ranges from 0 to 1. This value indicates the confidence level of the prediction and cannot be used to represent sentiment intensity.

12) AFFIN

Affin provides a score function, afn.score(text), which takes a text as input and returns a numeric value. If the returned value is greater than 0, the text is assigned a positive polarity; if it's lower than 0, the text is assigned a negative polarity.

13) STANZA (VIA CoreNLP)

Stanza employs a pipeline, nlp = stanza.Pipeline (lang= 'en', processors = 'tokenize,sentiment'), to download a dictionary. The input to this pipeline is the text, and it returns a document file. By using the sentiment function on each sentence in the document, the polarity of each text can be determined.

C. EVALUATION METRICS

In order to evaluate the accuracy of the packages in terms of both polarity and sentiment intensity, we require ground

truth data, which represents the actual values. As previously mentioned, the three product review datasets do not directly provide ground truth; instead, they offer summary texts that can function as proxies for ground truth. To achieve this, our process involves computing sentiment values for both the main text and the summary. Then, we establish a threshold of 0.2. If the difference between the sentiment values of the main text and the summary is below this threshold, we consider the sentiment value of the main text to be equal to that of the summary. This approach accounts for the improbability of having the same sentiment value for two distinct texts. Conversely, if the difference exceeds the threshold, we adjust the sentiment value of the main text to align with the sentiment value of the summary, taking into account the threshold.

For the Amazon and New York editorial datasets, sentiment intensity values are directly available, allowing us to use them without any modifications.

We evaluate the performance of the mentioned package at two distinct levels. The first level assesses sentiment values, while the subsequent level focuses on the classifier's accuracy in distinguishing positive and negative sentiments. This broader evaluation offers a more comprehensive understanding compared to solely examining the first level.

To assess accuracy of sentiment intensity, we compare predicted values (computed by the packages) with the actual values (ground truth). We employ two metrics for accuracy measurement: Root Mean Squared Error (RMSE) and R squared, as described below:

1) ROOT MEAN SQUARED ERROR (RMSE)

According to [23], the Root Mean Squared Error (RMSE) is computed using equation (1). RMSE is chosen as a measure because it effectively quantifies the maximum potential difference between two values. By taking the square root, RMSE aptly visualizes this difference. The RMSE value proximity to 0 indicates superior performance, while model error increases, its value increases.

The formula for RMSE is defined as follows:

$$\boldsymbol{RMSE} = \sqrt{\frac{1}{2}\sum_{l=1}^{l}\sum_{h=1}^{H}\left(\boldsymbol{O_{lh}} - \boldsymbol{Z_{lh}}\right)^2} \tag{1}$$

Here O represents the predicted output, Z represents the observed output, $l = 1 \ldots L$ denotes observation indices, and $h = 1 \ldots H$ represents output nodes. In our specific case, since we only have one output node for sentiment value (h = 1), the RMSE equation simplifies to:

$$\boldsymbol{RMSE} = \sqrt{\frac{1}{2}\sum_{l=1}^{l}\left(\boldsymbol{O_i} - \boldsymbol{Z_l}\right)^2} \tag{2}$$

$\boldsymbol{R^2}$. R-squared is a metric that gauges the quality of fit between the regression line and the data. It quantifies the proportion of the explained variation relative to the total variation, as depicted by equation (3). The R-squared value is bounded between 0 and 1, where a value of 1 denotes that the model comprehensively accounts for all variation around the mean. Conversely, a value of 0 implies that the model fails to explain any variation. However, in some special cases, it may return a negative value. We observed these instances in our experimental results, which we will describe later in this context. Essentially, an R-squared value within the 0 to 1 range signifies the model's accuracy in elucidating variation.

The formula for R-squared is articulated as follows:

$$\boldsymbol{R^2} = 1 - \frac{\boldsymbol{SSE}}{\boldsymbol{SST}} \tag{3}$$

where, SSE represents the sum of squared errors, and SST represents the total sum of squares.

Additionally, we can employ the Relative Square Error to calculate R-squared using equation (4).

$$\boldsymbol{R^2} = 1 - \boldsymbol{RSE} \tag{4}$$

To evaluate the accuracy of polarity classification (positive, negative, and neutral), we employ the accuracy metric, F-score metric, as well as recall and precision. This experiment was conducted using the IMDB and GOP dataset.

2) ACCURACY

As outlined in [24], the accuracy metric is determined using equation (5).

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN}} \tag{5}$$

In this formula, TP represents true positives (correctly classified positives), TN represents true negatives (correctly classified negatives), FP represents false positives (incorrectly classified positives), and FN represents false negatives (incorrectly classified negatives).

For this study, the classifier's result is classified as TP if the summary sentiment value is positive and the sentiment value of the entire text is also positive. The classifier's result is classified as FP if the sentiment value of the summary is negative and the sentiment value of the entire text is positive. The test result is classified as FN if the sentiment value of the summary is positive and the sentiment value of the entire text is negative. The test result is classified as TN if the sentiment value of the summary is negative and the sentiment value of the entire text is also negative.

3) F-SCORE

When calculating this measure, we adopt a multiclass approach, involving three sentiment polarity labels (positive, neutral, negative). As a result, we need to incorporate a parameter for the average attribute in the formula. In this context, we consider average parameter based on both micro and macro perspectives. As discussed by [25], given our work with an unbalanced dataset, the macro average is preferred. The F-score is computed using equation (6).

$$\boldsymbol{F - score} = \frac{\boldsymbol{2 * precision * Recall}}{\boldsymbol{Precision + Recall}} \tag{6}$$

To compute accuracy, F-score, precision, and recall, we employed the 'sklearn' library in Python.

TABLE 2. Experimental results on luxury products dataset (based on text and summary).

Package	RMSE	R^2
sentimentr	0.22	0.55
SA	0.19	0.53
qdap	0.42	0.26
syuzhet	0.50	-0.03
VS	0.35	-0.18
TextBlob	0.26	0.55
NLTK	0.35	-0.19
pattern	0.26	0.55

TABLE 3. Experimental results on software products dataset (based on text and summary).

Package	RMSE	R^2
sentimentr	0.25	0.52
SA	0.23	0.46
qdap	0.46	0.23
syuzhet	0.54	0.05
VS	0.37	-0.23
TextBlob	0.27	0.53
NLTK	0.38	-0.28
pattern	0.27	0.53

TABLE 4. Experimental results on industrial and scientific products dataset (based on text and summary).

Package	RMSE	R^2
sentimentr	0.23	0.54
SA	0.22	0.45
qdap	0.47	0.20
syuzhet	0.54	-0.22
VS	0.36	-0.32
TextBlob	0.28	0.48
NLTK	0.37	-0.37
pattern	0.28	0.48

TABLE 5. Experimental results on amazon dataset (based on gold standard).

Package	RMSE	R^2
sentimentr	0.40	0.22
SA	0.55	-0.46
qdap	0.44	0.06
syuzhet	0.74	-1.6
VS	0.37	0.32
TextBlob	0.39	0.24
NLTK	0.37	0.30
pattern	0.39	0.24

4) TIME COMPLEXITY

We also delve into the aspect of time complexity. In today's era of dealing with extensive data, having time-optimized algorithms is of paramount importance to ensure prompt output generation. Since internal algorithms' complexity information isn't readily available in terms of big O notation, this study assesses it based on execution time. To achieve this, we record the starting time before executing the desired code and capture the time again once the code finishes running.

TABLE 6. Experimental results on NY dataset (based on gold standard).

Package	RMSE	R^2
sentimentr	0.32	-0.02
SA	0.30	0.08
qdap	0.31	0.03
syuzhet	0.93	-7.48
VS	0.37	-0.35
TextBlob	0.35	-0.20
NLTK	0.37	-0.38
pattern	0.35	-0.2

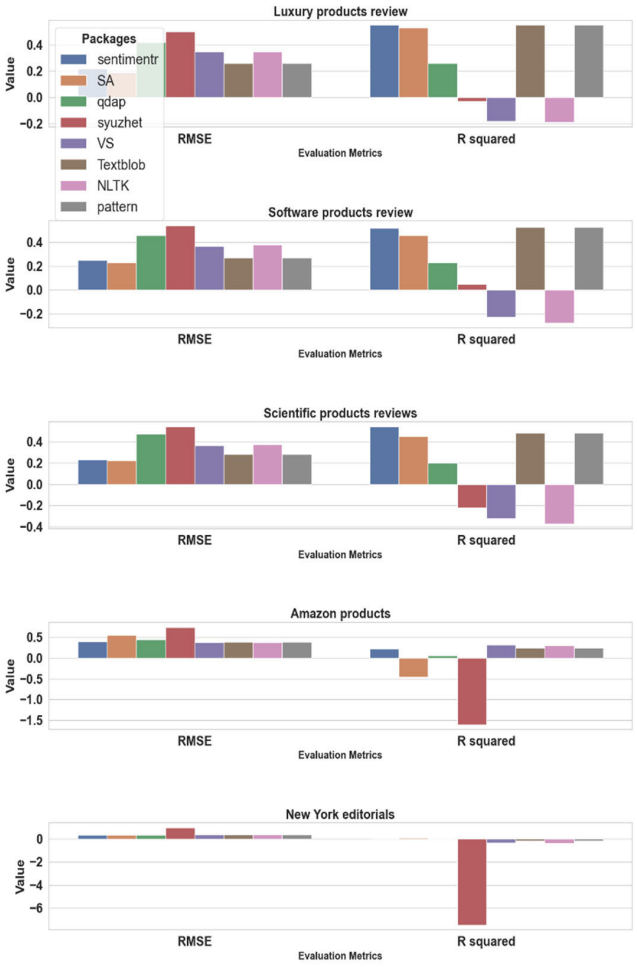

FIGURE 1. Bar chart depicting accuracy values in relation to sentiment intensity.

The difference between these two times is then regarded as the measure of time complexity.

IV. RESULTS

Initially, we calculated the sentiment intensity for three distinct product review datasets: luxury, software, and scientific products reviews. As previously mentioned, due to the unavailability of ground truth for these datasets, we rely on the text summaries as a proxy for ground truth. The results of these computations are presented in Tables 2, 3, and 4, corresponding to the aforementioned datasets.

TABLE 7. Accuracy of each package on IMDB dataset.

Package	Recall (macro)	Precision (macro)	F-score (macro)	Accuracy	Time (secs)
sentimentr	0.51	0.51	0.50	0.77	390.94
SA	0.68	0.68	0.68	0.68	6137.42
qdap	0.48	0.51	0.49	0.73	242.29
syuzhet	0.47	0.50	0.48	0.73	27.18
Rsentiment	0.24	0.40	0.28	0.38	1352.82
meanr	0.48	0.52	0.50	0.72	0.90
VS	0.48	0.50	0.48	0.74	11.57
Textblob	0.48	0.53	0.48	0.75	5.05
NLTK	0.48	0.50	0.48	0.74	67.56
Flair	0.93	0.93	0.93	0.93	743.65
Pattern	0.48	0.53	0.48	0.75	11.48
Affin	0.48	0.51	0.49	0.74	21.63
Stanza (via CoreNLP)	0.33	0.54	0.41	0.50	5592

These outcomes showcase the performance of eight widely recognized packages in terms of accuracy. We evaluate their performance using the RMSE and R-squared metrics. It is important to note that we specifically selected these packages because, among the various packages examined in this study, they are the only ones providing sentiment intensity as a continuous value.

In the second experiment, we employ two datasets: Amazon and GOP debate datasets, both of which offer ground truth values. Accuracy results for these two datasets, using RMSE and R-squared as evaluation metrics, are presented in Tables 5 and 6, respectively.

To facilitate a more comprehensive comparison, we have illustrated the results extracted from Tables 2, 3, 4, 5, and 6 in Fig 1.

To assess the accuracy of polarity classification within the packages, an experiment was undertaken on two datasets: IMDB and GOP debate. The IMDB dataset is widely recognized and commonly used in sentiment analysis, whereas the GOP debate dataset is less familiar. The outcomes of these experiments are presented in Tables 7 and 8, showcasing the results for IMDB and GOP datasets, respectively.

To assess accuracy, we utilize two metrics: F-score and accuracy. Furthermore, for error analysis, precision and recall are calculated. Additionally, these two datasets are employed to analyze time complexity.

TABLE 8. Accuracy of each package on GOP debate twitter sentiment dataset.

Package	Recall (macro)	Precision (macro)	F-score (macro)	Accuracy	Time (secs)
sentimentr	0.47	0.50	0.45	0.48	565.59
SA	0.38	0.29	0.25	0.33	9206.48
qdap	0.50	0.48	0.43	0.44	144.62
syuzhet	0.50	0.46	0.43	0.46	20.01
Rsentiment	0.41	0.42	0.33	0.34	646.76
meanr	0.49	0.47	0.42	0.42	4.80
VS	0.49	0.46	0.43	0.44	1.65
Textblob	0.48	0.44	0.37	0.37	2.59
NLTK	0.49	0.46	0.43	0.44	2.50
Flair	0.48	0.33	0.38	0.55	291.94
Pattern	0.45	0.44	0.37	0.37	2.63
Affin	0.47	0.45	0.41	0.42	5.97
Stanza (via CoreNLP)	0.47	0.38	0.26	0.30	2192.58

FIGURE 2. Bar chart depicting accuracy of packages in terms of polarity classification, evaluated using accuracy and F-score metrics, for IMDB and GOP datasets.

For enhanced comparison, the results extracted from Tables 7 and 8 are presented in Fig 2. Additionally, Fig 3 illustrates the execution time of packages when running on both the IMDB and GOP datasets.

V. DISCUSSION

A. SENTIMENT INTENSITY ANALYSIS

Firstly, it is important to note that certain packages (meanr, Rsentiment, Flair, stanza, and Affin) do not offer continuous

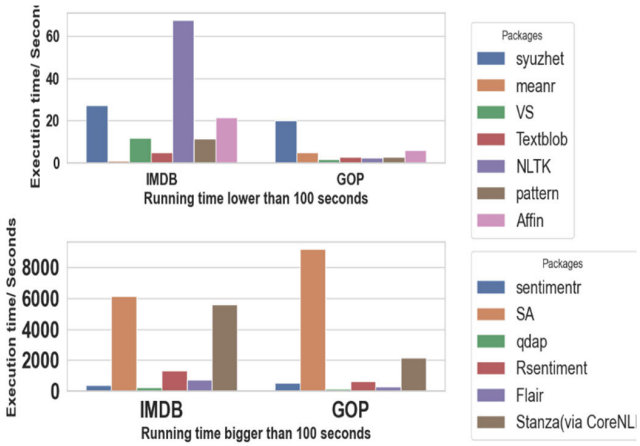

FIGURE 3. Running time comparison for each package on IMDB and GOP datasets. The left section displays packages with a running time lower than 100 seconds, while the right section showcases packages with a running time exceeding 100 seconds.

sentiment intensity scores. Therefore, this analysis focuses on 'sentimentr', 'SentimentAnalysis', 'qdap', and 'meanr' from R, as well as 'VaderSentiment', 'Textblob', 'NLTK', and 'pattern' from Python.

Regarding the two datasets with a gold standard, a contrast emerged. For the Amazon dataset, Python libraries generally outperformed R packages, with 'VaderSentiment' leading Python with an accuracy of 0.37 for RMSE and 0.32 for R^2. Among R packages, 'sentimentr' achieved high accuracy, scoring 0.4 for RMSE and 0.22 for R^2. On the other hand, Table 6 showcases results for the New York dataset. Here, most R packages outperformed Python libraries. The 'SentimentAnalysis' package stood out, recording the highest accuracy with 0.30 for RMSE and 0.08 for R^2. Conversely, 'syuzhet' demonstrated the lowest accuracy for both datasets. Comparing Tables 5 and 6 reveals that the performance of 'VaderSentiment' and 'NLTK' are the same across both datasets. A similar observation applies to 'TextBlob' and 'pattern' libraries. Additionally, this comparison highlights a significant observation: across all packages, accuracy values in both metrics (RMSE and R Squared) are consistently higher for the Amazon dataset compared to the NY dataset. This finding strengthens the hypothesis that existing packages and libraries might be overfitted to the dataset.

To maintain an unbiased approach, we conducted experiments on three additional datasets: luxury products, software products, and scientific products review datasets. As discussed in section III-A, the summary text field is treated as the proxy ground truth. Tables 2, 3, and 4 present experimental results for these datasets. The 'sentimentr' package displays relatively low RMSE values (0.22, 0.25, and 0.23 respectively for the mentioned datasets) and moderate to high R^2 values across all three datasets (0.55, 0.52, and 0.54 respectively). This suggests that 'sentimentr' offers accurate and well-fitted sentiment intensity predictions for product reviews. The SentimentAnalysis package in R also performs reasonably well with relatively low RMSE and moderate to high R^2 values across the datasets. Among the Python libraries, Textblob and pattern exhibit consistent performance across the datasets, surpassing the other two libraries with comparable RMSE and R^2values. In summary, the experimental results indicate that 'sentimentr' in R consistently delivers accurate and well-fitted sentiment intensity predictions for product review datasets.

B. POLARITY ANALYSIS

Flair emerges as the leader in sentiment classification accuracy for both datasets, achieving impressive scores of 0.93 and 0.55 for IMDB and GOP respectively. However, an inherent challenge arises when using the accuracy metric in this context. Designed for binary classification, it doesn't accurately support a multiclass problem. To address this, we incorporate the F-score measure for a more reliable comparison.

Upon calculating the F-score across all packages, we find that Flair excels as the best classifier for IMDB, boasting a remarkably high score of 0.93. However, its performance drops significantly to around 0.38 on the GOP dataset. A similar trend emerges for the SentimentAnalysis package, demonstrating strong performance (0.68) on IMDB but faltering with a low score of 0.25 on the GOP dataset.

Comparing various packages, 'sentimentr' consistently delivers good performance. It achieves an F-score of 0.50 for IMDB and 0.45 for GOP, along with accuracy values of 0.77 and 0.48 respectively. The F-score difference between IMDB and GOP is negligible for Vadersentiment, standing at 0.48 and 0.43 respectively.

On the contrary, the poorest performance, indicated by both F-score and accuracy, comes from 'Rsentiment' with values of 0.28 and 0.38 for IMDB, and from SentimentAnalysis with an F-score of 0.25 for GOP. Additionally, in terms of accuracy, Stanza demonstrates the weakest performance for the GOP dataset.

In conclusion, discerning which programming language offers superior accuracy is challenging. The accuracy value seems to depend largely on the specific dataset.

C. TIME COMPLEXITY

Significant variability exists in the time complexity among different packages for sentiment analysis. The duration of sentiment analysis can hinge on multiple factors, including the intricacy of the underlying algorithms, dataset size, and implementation efficiency. We conducted these analyses on a Core i7 PC equipped with 16 GB RAM, running Windows 10. The R version used was 4.31, and Python version 3.11.0.

To ensure accurate time measurement, we computed the execution time for each individual text before summing up the times. This approach is taken due to variations in how programming languages handle vectors, with some packages encountering errors when processing vectors.

Table 7 and 8 showcase the execution times for each package and library. It's noteworthy that the IMDB dataset

comprises 5550 texts and 716349 words, while the GOP dataset encompasses 9425 texts and 153221 words.

For the IMDB dataset, the 'meanr' package demonstrates the swiftest execution time at 0.9 seconds. The second fastest library is TextBlob with a time of 5.05 seconds. In contrast, the slowest performer in this dataset is SentimentAnalysis, requiring a substantial 6137.42 seconds. In the case of the GOP dataset, 'vadersentiment' takes the lead with an execution time of 1.65 seconds, while SentimentAnalysis again trails, demanding 9206.48 seconds. Regardless of the dataset, it's evident that SentimentAnalysis consistently ranks as the slowest package.

Of interest is the balance achieved by the 'meanr' package. Despite its high execution speed, it maintains acceptable performance with accuracy values of 0.72 (IMDB) and 0.42 (GOP), as well as F-score of 0.50 and 0.42 respectively, all achieved within approximately 4.80 seconds. In contrast, stanza, while sluggish, offers lower accuracy. With an execution time of 5592 seconds (IMDB) and 2192.58 seconds (GOP), it achieves accuracy levels of 0.50 and 0.30 respectively, accompanied by F-score of 0.41 and 0.26.

The experimental outcomes reveal a marked decrease in execution time for all Python libraries on the GOP dataset. Notably, NLTK experienced a speed increase of up to 27 times, while VaderSentiment demonstrated up to 7 times faster execution. Similar trends held for other libraries, varying from 2 to 4 times faster. Among R packages, this phenomenon was observed in 'qdap', 'syuzhet', and 'Rsentiment', albeit to a lesser degree compared to Python libraries. This discovery underscores the distinction between the word-level processing of Python libraries and the sentence-level approach of R packages. This discrepancy is reflected in the lower word count of the GOP dataset compared to IMDB, despite the GOP dataset containing more text entries.

D. OVERFITTING

Overfitting, a biased behavior observed in supervised learning, manifests when a model becomes excessively tailored to specific training data, leading to poor performance on test data. In this study, we identify signs of overfitting in lexicon and rule-based methods used for sentiment analysis.

Initially, the experimental results on IMDB and GOP debate datasets reveal distinct behaviors of packages, as outlined below. The flair library notably excels on the IMDB dataset, boasting an impressive 93% accuracy and F-score. However, when applied to the GOP dataset, its accuracy drops to 0.55, accompanied by a F-score of 0.38%, which is one of the lowest among Python libraries. Investigating this phenomenon, we [26], which reports that flair's training is predominantly based on the IMDB dataset. This alignment demonstrates flair's overfitting on the IMDB dataset.

Another noteworthy concern arises from the accuracy variation between the Amazon and NY datasets versus the three product review datasets. Python libraries exhibit higher accuracy levels on the Amazon dataset compared to R packages, yet this disparity does not extend to the NY dataset. This discrepancy calls for further exploration, as [12] mentioned that the Vader lexicon is aptly designed for social media texts. Interestingly, even though 'sentimentr' boasts accuracy in the three product review datasets, it retains relative accuracy when compared to other packages in the NY dataset.

Moreover, the experimental results highlight a consistent trend: across programming languages, all packages and libraries exhibit low performance on lesser-known datasets. This trend supports the idea that these tools are typically trained on specific datasets, limiting their effectiveness when faced with unfamiliar data.

E. ERROR ANALYSIS

We conducted an analysis of false positive and false negative errors, pivotal components in the computation of precision and recall for both the F-score and accuracy metrics. The calculation of precision and recall follows equations (7) and (8) [24].

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

Precision and recall values for each package are presented in Tables 7 and 8. The 'macro' parameter is employed to compute these values. When examining the experiment results on the IMDB dataset, we observe that recall values for most packages/libraries are lower than their corresponding precision values. Moreover, two packages exhibit equal precision and recall. This pattern implies that in these cases, there is a prevalence of incorrectly classified negatives ($FN > FP$). Consequently, across various packages, a tendency exists to classify texts as negative within the IMDB dataset.

Conversely, analysis of the GOP dataset indicates a different scenario. With the exception of 'sentimentr' and 'Rsentiment', recall values exceed precision values for all libraries and packages. This configuration points to a prevalence of incorrectly classified positives ($FP > FN$). As a result, the trend across Python packages and most R packages within the GOP dataset leans toward classifying texts as positive.

F. CASE STUDIES

To conduct firsthand research into sentiment intensity provided by various packages, we randomly selected seven texts along with their summarized versions from the Luxury dataset. Table 9 illustrates the outcomes of these experiments.

Let's delve into the analysis of the first review text from Table 9. For the sentence "I have been using this product for many years. Although it is expensive, it is the only toothpaste I use," the sentimentr package yields a score of -0.09. Similarly, 'syuzhet' and 'qdap' assess the sentiment as negative with values of -0.12 and -0.22 respectively. In contrast, all Python libraries categorize this sentence as neutral. This

TABLE 9. Case studies from luxury dataset.

Review text (R)	Summary Text (S)	sentimentr		SA		qdap		syuzhet		VS		Textblob		NLTK		pattern	
		R	S	R	S	R	S	R	S	R	S	R	S	R	S	R	S
I have been using this product for many years. Although it is expensive, it is the only toothpaste I use.	*Excellent Product*	-0.09	0.7	0.0	0.5	-0.22	0.70	-0.12	1	0.0	0.57	0.0	1.0	0.0	0.57	0.0	1.0
Really pretty color, but not good for a yellower skin-tone. Gave it to my mom who is more of a pink skin tone. Very blue undertones	*Pretty and subtle*	-0.04	0.43	0.07	0.0	0.13	0.57	0.5	0.75	-0.21	0.49	0.06	-0.04	-0.21	0.49	0.06	-0.04
Color is not as depicted in picture. I thought it was a subtle coral, but it turned out to be a bright tangerine orange.	*Wrong color in pic*	0.14	-0.37	0.0	-0.33	0.20	-0.5	0.25	-0.75	0.59	-0.47	0.18	-0.5	0.59	-0.47	0.18	-0.5
Overall, I like this product. It is very creamy and easy to apply. Unfortunately, it was a bit darker than I was expecting but that was my fault.	*Good!*	0.09	0.75	-0.09	1	-0.49	1	-0.26	0.75	-0.34	0.49	0.03	0.87	-0.34	0.49	0.03	0.87
We love the smell of this stuff! Keeps our babies skin soft and never has irritated it!	*Wonderful product!*	0.07	0.53	0	0	0	0.70	0.3	0.75	0.8	0.61	0.37	1	0.8	0.61	0.37	1
These razor blades are way overpriced. I have gotten a much better shave from blades that are much cheaper. The Astra or Derby Blades in particular give a better smoother shave and are much cheaper.	*Way too expensive ...*	0.59	-0.14	0.09	0	0.94	-0.57	0.56	-0.25	0.70	0.0	0.31	-0.5	0.70	0.0	0.31	-0.5
This product holds the fibers in place for several days as I do not shampoo my hair everyday and only occasionally have to add fibers to certain spots.	*Great for hold*	0	0.28	-0.07	0.5	0	0.57	0.8	0.5	0.27	0.62	0.00 2	0.8	0.27	0.62	0.002	0.8

example underscores a drawback of lexicon-based methods like 'qdap'; they struggle when faced with sentences devoid of emotionally charged words. It's evident that the first sentence carries a positive sentiment. Interestingly, the summary text ''Excellent Product'' receives an intensity value of 0.5 from 'SentimentAnalysis'. However, this value is a misrepresentation, as the sentiment is unmistakably positive. 'VaderSentiment' and 'NLTK' fare only marginally better with a value of 0.57 each. Notably, 'Pattern', 'TextBlob', and 'syuzhet' align better with a more fitting sentiment value of 1.

The second issue with these packages arises from their incapability to detect contradictions within sentences. In the second sentence of the text, the reviewer expresses contentment despite acknowledging the product's costliness.

Turning to the second review text, ''Really pretty color, but not good for a yellower skin-tone. Gave it to my mom who is more of a pink skin tone. Very blue undertones,'' we see that 'SentimentAnalysis', 'qdap', 'syuzhet', 'Textblob', and 'pattern' assign a positive sentiment value. This interpretation is accurate, as the text is indeed positive. Despite mentioning that the product isn't suitable for a yellower skin-tone, the reviewer ultimately expresses a recommendation. Conversely, 'sentimentr', 'VaderSentiment', and 'NLTK' perceive this text as negative. A significant limitation of these packages and libraries lies in their inability to comprehend the intricate relationships between words in a text. Intriguingly, 'SentimentAnalysis', 'pattern', and 'TextBlob' attribute a negative or neutral sentiment to the summary text ''Pretty and subtle'' which contradicts the actual sentiment conveyed.

Moving on to the third review text, ''Color is not as depicted in picture. I thought it was a subtle coral, but it turned out to be a bright tangerine orange.'' Despite the text's negative nature, all packages and libraries misclassify it as a positive review. 'SentimentAnalysis' stands out slightly by providing a more neutral score of 0. It appears that these packages computed sentiment values based on terms such as ''subtle'' and ''bright.'' Examining the summary text, ''Wrong color in pic,'' reveals that while all packages assign a reasonable negative value to it, there exists notable variance among their interpretations.

The fourth case underscores the influence of intensifiers on sentiment values. The text consists of three sentences: ''Overall, I like this product,''''It is very creamy and easy to apply,'' and ''Unfortunately, it was a bit darker than I was expecting, but that was my fault.'' 'Sentimentr' assigns sentiment values of 0.22, 0.50, and −0.70, respectively. The presence of the term ''very'' in the second sentence contributes to it being twice as positive as the first sentence. Additionally, the third sentence doesn't convey strong negative sentiment towards the product, and a score of −0.70 seems an exaggeration. Interestingly, aside from 'Sentimentr', only 'TextBlob' and 'pattern' interpret it positively; the other packages categorize it as negative.

Moving to the fifth review text, ''We love the smell of this stuff! Keeps our baby's skin soft and never has irritated it!'' 'SentimentAnalysis' and 'qdap' consider it neutral, while 'sentimentr' assigns a low value of 0.07, also interpreted as neutral. This text, conveying strong positivity, is best recognized by 'VaderSentiment' and 'NLTK,' both assigning

high positive scores of 0.8. The corresponding summary text, ''Wonderful product!'', mirrors this sentiment, but 'pattern' and 'TextBlob' uniquely assign it the highest positive value of 1, while 'SentimentAnalysis' views it neutrally.

Exploring the sixth example once again underscores the limitations of existing packages within this study in accurately discerning the reviewer's sentiment toward different products. In this text, the reviewer expresses negative sentiment about a specific type of razor in the first sentence, followed by positive sentiment about other types of razors in the second sentence. The overall sentiment of the text leans toward negativity regarding the product under review. Astonishingly, all existing packages consider this text as positive, highlighting a significant flaw in lexicon and rule-based approaches. Analyzing the summary text, ''Way too expensive,'' which conveys a negative sentiment, reveals an intriguing inconsistency. 'SentimentAnalysis,' 'VaderSentiment,' and 'NLTK' categorize it as neutral, while other packages interpret it as positive, albeit with varying intensity.

The case studies provided in Table 9 starkly expose a major deficiency in the existing packages under scrutiny. Not only do they struggle to accurately compute sentiment intensity in most cases, but they also fall short in accurately classifying the polarity of the text. This further emphasizes the complexity and challenges inherent in sentiment analysis.

VI. RESEARCH DIRECTIONS

According to this comprehensive study of lexicon and rule based packages in R and Python programming languages, the following research directions are highlighted:

- A similar approach to evaluate the accuracy of machine learning methods in both programming languages is highly valuable, particularly to investigate the occurrence of overfitting in ML methods.
- Contributing to similar studies, finding another creative way to create ground truth data is crucial, while annotator-based approaches are challenging and error-prone.
- The study reveals that the 'sentimentr' package is well-suited among the 13 studied packages. However, a comprehensive study on its background and internal functions would be highly valuable. This would provide insights into using this package with different datasets.

VII. CONCLUSION

This study centered on a comprehensive evaluation of six rule and lexicon-based packages in R and seven similar packages in Python. The focus was on comparing sentiment intensity across these packages using metrics such as RMSE and R squared. However, obtaining ground truth data for sentiment intensity proves challenging, given the complexity of assigning continuous values through annotators. As an alternative, we leveraged a new approach that considered the summary of text as a form of ground truth, which we deemed necessary due to the consistent source of the two datasets employed. Additionally, we assessed polarity classification accuracy using the F-score accuracy metric.

Our findings suggest that the 'sentimentr' package demonstrates relatively strong performance across most datasets. Interestingly, Python libraries showcase better accuracy than R packages on the Amazon dataset, developed by the creator of 'Vadersentiment'. However, their performance deteriorates significantly on the GOP dataset, as indicated by the R squared value. It's noteworthy that packages, in general, exhibit a lower R squared value.

Furthermore, our study uncovers evidence of overfitting, with most libraries excelling on well-known datasets but struggling on unfamiliar ones. The results, supported by running time observations, indicate that Python libraries predominantly operate at the word level, while R packages function at the sentence level. Notably, Python libraries showcase notably faster processing speeds compared to R packages, highlighting their efficiency.

In conclusion, this study contributes valuable insights into the strengths and limitations of sentiment analysis packages. By comparing performance metrics, addressing overfitting concerns, and analyzing computational efficiency, we shed light on the complexities of sentiment analysis within the context of rule and lexicon-based approaches.

REFERENCES

[1] M. M. Mostafa, ''More than words: Social networks' text mining for consumer brand sentiments,'' *Expert Syst. Appl.*, vol. 40, no. 10, pp. 4241–4251, Aug. 2013.

[2] T. H. Nguyen, K. Shirai, and J. Velcin, ''Sentiment analysis on social media for stock movement prediction,'' *Expert Syst. Appl.*, vol. 42, no. 24, pp. 9603–9611, Dec. 2015.

[3] P. Mehta, S. Pandya, and K. Kotecha, ''Harvesting social media sentiment analysis to enhance stock market prediction using deep learning,'' *PeerJ Comput. Sci.*, vol. 7, p. e476, Apr. 2021.

[4] M. Elshendy, A. F. Colladon, E. Battistoni, and P. A. Gloor, ''Using four different online media sources to forecast the crude oil price,'' *J. Inf. Sci.*, vol. 44, no. 3, pp. 408–421, Jun. 2018.

[5] M. Qorib, T. Oladunni, M. Denis, E. Ososanya, and P. Cotae, ''COVID-19 vaccine hesitancy: Text mining, sentiment analysis and machine learning on COVID-19 vaccination Twitter dataset,'' *Expert Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118715.

[6] G. Ganczewski and D. Jemielniak, ''Twitter is garbage: A thick big data exploration of #zerowaste hashtag on Twitter in relation to packaging and food packaging materials,'' *Packag. Technol. Sci.*, vol. 35, no. 12, pp. 893–902, Aug. 2022, doi: 10.1002/pts.2685.

[7] A. M. Górska, K. Kulicka, and D. Jemielniak, ''Men not going their own way: A thick big data analysis of #MGTOW and #Feminism tweets,'' *Feminist Media Stud.*, vol. 23, no. 8, pp. 3774–3792, Nov. 2023.

[8] A. Mahmoudi, ''Identifying biased users in online social networks to enhance the accuracy of sentiment analysis: A user behavior-based approach,'' 2021, *arXiv:2105.05950*.

[9] B. Verma and R. S. Thakur, ''Sentiment analysis using lexicon and machine learning-based approaches: A survey,'' in *Proc. Int. Conf. Recent Advancement Comput. Commun.*, in Lecture Notes in Networks and Systems. Singapore: Springer, 2018, pp. 441–447.

[10] R. Srivastava, P. K. Bharti, and P. Verma, ''Comparative analysis of lexicon and machine learning approach for sentiment analysis,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 3, pp. 71–77, 2022. [Online]. Available: https://www.academia.edu/download/87358181/Paper_12-Comparative_Analysis_of_Lexicon_and_Machine_Learning_Approach.pdf

[11] H. Nguyen, A. Veluchamy, M. Diop, and R. Iqbal, ''Comparative study of sentiment analysis with product reviews using machine learning and lexicon-based approaches,'' *SMU Data Sci. Rev.*, vol. 1, no. 4, p. 7, 2018.

[12] C. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," in *Proc. ICWSM*, May 2014, vol. 8, no. 1, pp. 216–225.
[13] C. Musto, G. Semeraro, and M. Polignano, "A comparison of lexicon-based approaches for sentiment analysis of microblog posts," in *Proc. DART@ AI* IA*, 2014, pp. 59–68. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f0401ab7579c94dca7fdc5fba6d8f1665d155a58#page=66
[14] M. Naldi, "A review of sentiment computation methods with R packages," 2019, *arXiv:1901.08319*.
[15] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguistics*, vol. 37, no. 2, pp. 267–307, Jun. 2011. [Online]. Available: http://www.jstor.org/stable/20182753%0D
[16] M. A. Al-Shabi, "Evaluating the performance of the most important Lexicons used to sentiment analysis and opinions mining," *Int. J. Comput. Sci. Netw. Secur.*, vol. 20, no. 1, p. 1, 2020.
[17] L. Young and S. Soroka, "Affective news: The automated coding of sentiment in political texts," *Political Commun.*, vol. 29, no. 2, pp. 205–231, Apr. 2012.
[18] W. van Atteveldt, M. A. C. G. van der Velden, and M. Boukes, "The validity of sentiment analysis: Comparing manual annotation, crowd-coding, dictionary approaches, and machine learning algorithms," *Commun. Methods Measures*, vol. 15, no. 2, pp. 121–140, Apr. 2021.
[19] T. Shaik, X. Tao, C. Dann, H. Xie, Y. Li, and L. Galligan, "Sentiment analysis and opinion mining on educational data: A survey," *Natural Lang. Process. J.*, vol. 2, Mar. 2023, Art. no. 100003.
[20] J. Ni. (2018). *Amazon Review Data 2018*. Accessed: Feb. 12, 2023. [Online]. Available: https://nijianmo.github.io/amazon/index.html
[21] (Feb. 2021). *Dictionary-Based Sentiment Analysis [R Package SentimentAnalysis Version 1.3-4]*. Accessed: Jul. 15, 2023. [Online]. Available: https://cran.r-project.org/web/packages/SentimentAnalysis
[22] S. Feuerriegel and N. Proellochs. (Feb. 18, 2021). *SentimentAnalysis Vignette*. Accessed: Jul. 15, 2023. [Online]. Available: https://cran.r-project.org/web/packages/SentimentAnalysis/vignettes/ SentimentAnalysis.html
[23] F. Günther and S. Fritsch, "NeuralNet: Training of neural networks," *R J.*, vol. 2, no. 1, p. 30, 2010.
[24] H. Dalianis, "Evaluation metrics and evaluation," in *Clinical Text Mining: Secondary Use of Electronic Patient Records*, H. Dalianis, Ed. Cham, Switzerland: Springer, 2018, pp. 45–53.
[25] (Jun. 19, 2022). *Understanding Micro, Macro, and Weighted Averages for Scikit-Learn Metrics in Multi-Class Classification With Example*. Amir Masoud Sefidian—Sefidian Academy. Accessed: Aug. 12, 2023. [Online]. Available: http://iamirmasoud.com/2022/06/19/understanding-micro-macro-and-weighted-averages-for-scikit-learn-metrics-in-multi-class-classification-with-example/
[26] A. McFarland. (Jul. 4, 2022). *10 Best Python Libraries for Sentiment Analysis*. Accessed: Aug. 13, 2023. [Online]. Available: https://www.unite.ai/10-best-python-libraries-for-sentiment-analysis/

AMIN MAHMOUDI received the Ph.D. degree from The National University of Malaysia. He is currently a Postdoctoral Researcher with the Management in Networked and Digital Societies (MINDS) Department, Kozminski University, Warsaw, Poland. His research interest includes online social network analysis.

DARIUSZ JEMIELNIAK is currently a Full Professor and the Head of the Management in Networked and Digital Societies (MINDS) Department, Kozminski University, and the Faculty Associate with the Berkman-Klein Center for Internet and Society, Harvard University. He is a member of the Polish Academy of Sciences. His current research interests include disinformation and bot detection. He also serves on the Wikimedia Foundation Board of Trustees and the Vice-President for the Polish Academy of Sciences.

LEON CIECHANOWSKI is currently a Cognitive Scientist, a Philosopher, and a Culture Science Expert. He is also an Assistant Professor with the Management in Networked and Digital Societies (MINDS) Department, Kozminski University, and the Psychology Department with the SWPS University of Social Sciences and Humanities, Warsaw. He is involved in the study of chatbots, robots and wearables in cooperation with MIT. His research interests include human–computer interaction, phenomenology, and sense of agency/control.