Classifying #LongCovid tweets into different categories, such as symptoms, treatments, personal experiences, or scientific research

Marija Saveska - 201062

Faculty of computer science and engineering
UKIM
Skopje, Republic of North Macedonia
marija.saveska@students.finki.ukim.mk

***Abstract* — In recent years, the exponential growth of social media platforms has led to an influx of information sharing on a global scale. Among these platforms, Twitter has emerged as a significant channel for people to express their thoughts, experiences, and opinions on various topics. One area that has gained significant attention is the discussion surrounding Long Covid, a condition where individuals experience persistent symptoms following a Covid-19 infection. With the overwhelming volume of data being generated on Twitter, there is a growing need to effectively categorize and analyze this information to extract valuable insights.**

This paper presents an approach to address this challenge through text classification techniques. The primary objective is to classify #LongCovid-related tweets into distinct categories such as symptoms, treatments, personal experiences, and scientific research. By doing so, we aim to provide a structured framework that organizes the diverse information shared on Twitter. Through this classification process, valuable insights can be gained from understanding the prevalence of different categories, identifying common symptoms, exploring various treatments, and discerning scientific research trends.

The proposed text classification framework involves the utilization of machine learning algorithms and natural language processing techniques. In the paper the data preprocessing steps, feature extraction methods, and the training of classification models are discussed. Additionally, the challenges posed by the informal and noisy nature of Twitter text are addressed, and strategies are explored to enhance the accuracy and robustness of our classification approach.

In conclusion, this paper underscores the importance of effectively classifying #LongCovid-related tweets into relevant categories. By harnessing text classification techniques, we can streamline the analysis of Twitter data, thereby aiding researchers, healthcare professionals, and policymakers in gaining a deeper understanding of the Long Covid phenomenon. Through this process, we pave the way for informed decision-making and improved insights in the context of post-Covid-19 recovery.

Key words* — *Twitter, #LongCovid, symptoms, treatments, scientific research, personal experiences, machine learning, text processing;

I. INTRODUCTION

Twitter has emerged as a prominent medium for real-time discussions. This instant and widespread connectivity has led to the creation of vast digital landscapes characterized by an abundance of unstructured text data. Within this digital realm, individuals often express their thoughts, emotions, and opinions on diverse topics, including health-related matters.

This paper is motivated by the necessity to harness the power of text classification techniques to navigate the vast landscape of #LongCovid-related tweets. The overarching goal is to categorize these tweets into distinct classes such as symptoms, treatments, personal experiences,

and scientific research. This classification endeavor holds the potential to unravel valuable insights buried within the massive pool of Twitter data.

Text classification, a subfield of natural language processing (NLP) and machine learning, plays a pivotal role in this pursuit. By employing computational algorithms, the process of categorizing tweets based on their content is automated, enabling us to discern prevalent symptoms, explore emerging treatments, and identify narratives that resonate with personal experiences. The importance of this endeavor is underscored by the increasing relevance of Long Covid in the global health discourse. The availability of structured information can aid healthcare professionals, researchers, and policymakers in better understanding the nuances of this condition and making informed decisions.

II. METHODOLOGY

In this section, we outline the methodology employed to tackle the task of classifying #LongCovid-related tweets into distinct categories. This encompasses the acquisition of the dataset, preprocessing steps to handle challenges like missing values and unclean text, the tools used for implementation, and the step-by-step approach to text classification.

2.1. Data Collection and Preprocessing

The foundation of the methodology used rests upon a dataset sourced from Kaggle. This dataset, containing columns for date, user, and tweets, was obtained to form the basis of the analysis. However, data preprocessing was imperative due to the inherent variability of user-generated content. The dataset presented challenges in the form of missing values (NaN) and unclean text within tweets.

To address these issues, data preprocessing techniques were employed. Missing values were handled, ensuring that the integrity of the dataset was maintained. Additionally, the text within the tweets was subjected to a series of cleaning steps, including the removal of special characters, URLs, and irrelevant symbols. This cleaning process was pivotal to ensure that the classification models operated on standardized and consistent input.

The dataset mentioned above is https://www.kaggle.com/datasets/matt0922/longcovid-tweets?select=lc2022.csv.

2.2. Tools and Implementation

To carry out the classification tasks, Google Colaboratory—a cloud-based platform that provided computational resources and seamless integration with Python libraries was used. Python, being a versatile programming language, was employed to write the necessary code for data manipulation, preprocessing, and classification.

2.3. Text Classification Approach

The text classification approach consisted of several crucial steps:

Tokenization: The textual data was tokenized, breaking down tweets into individual words or tokens. This facilitated the transformation of text into a format suitable for analysis.

Manual Classifying: A significant challenge with this endeavor was the absence of pre-labeled data for training. Thus, a manual classification process was initiated, involving the categorization of tweets into the desired categories—symptoms, treatments, personal experiences, and scientific research. This labeled dataset formed the foundation for training and evaluation.

Support Vector Classifier (SVC): Leveraging machine learning algorithms, we implemented a Support Vector Classifier (SVC) to classify tweets based on the manually assigned labels. SVC, a well-established algorithm for text classification, was trained using the labeled dataset to see how successful was the manual classifying.

2.4. Evaluation Methods

The efficacy of the classification approach was evaluated through rigorous assessment. Standard evaluation metrics were employed such as precision, recall, and F1-score to quantify the performance of the model. These metrics gauged the accuracy of the classification results, ensuring that the model's predictive capabilities were effectively measured.

III. DATA ANALYSIS

3.1. Data preparation

Effective data preparation is the cornerstone of any data analysis and modeling endeavor. In the context of classifying #LongCovid-related tweets, a meticulous data preparation process was instrumental in ensuring the quality and consistency of our dataset.

3.1.1. Handling Missing Values

Upon acquiring the dataset, one of the primary challenges encountered was the presence of missing values (NaN) within certain rows. Addressing this issue was crucial to maintain data integrity and prevent potential biases during classification. The strategy employed to mitigate this challenge involved the removal of rows containing NaN values. By utilizing the 'dropna()' function, rows with incomplete data were eliminated, creating a clean and consistent dataset for subsequent analysis.

3.1.2. Text Cleaning

Tweets obtained from social media platforms are often laden with noise, encompassing hashtags, mentions, URLs, and emojis that do not contribute to the classification task. To ensure that our text classification models operated on meaningful content, a comprehensive text cleaning process was undertaken.

The process consisted of a series of essential cleaning steps, each designed to address specific types of noise:

Removing Stop Words: Stop words, commonly occurring words with little contextual value, were removed to focus on the core content of the tweets.

Fig. 1 Stop words

```
stop_words=stopwords.words("english")
def remove_stop_words(text):
    return " ".join([word for word in text.split(" ") if word not in stop_words])
```

Eliminating URLs: URLs were stripped from tweets using regular expressions, ensuring that web links did not interfere with the text classification process.

Fig. 2 HTTP Urls

```
def remove_urls(text):
    return re.sub('((www\.[^\s]+)|(https?://[^\s]+))'," ",text)
```

Handling Hashtags and Mentions: Hashtags and mentions were removed, as they carried minimal relevance in the context of category classification.

Fig. 3 Hashtags

```
hashtag_re=re.compile(pattern='#[\w\d]+')
def remove_hashtag(text:str)->str:
    return hashtag_re.sub(repl="",string=text)
```

Fig. 4 Mentions

```
mention_re =re.compile('@\w+')
def remove_mention(text):
    return mention_re.sub(repl="",string=text)
```

Filtering Numbers: Numbers, which often held limited semantic value, were excluded from the cleaned text.

Fig. 5 Numbers

```
number_re=re.compile('\d+')
def remove_numbers(text):
    return number_re.sub(repl='',string=text)
```

Handling Multiple White Spaces: Extra spaces were removed, enhancing the uniformity of the text.

Fig. 6 Multiple whitespaces

```
multiple_space_re =re.compile('\s{2,}')
def remove_multiple_whitespace(text):
    return multiple_space_re.sub(repl=' ',string=text)
```

Removing Emojis: Emojis, although expressive, were stripped from the text to prevent their influence on classification.

Fig. 7 Emojis

```
def remove_emojis(text):
    emoji_pattern = re.compile("["
                               u"\U0001F600-\U0001F64F"  # emoticons
                               u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                               u"\U0001F680-\U0001F6FF"  # transport & map symbols
                               u"\U0001F700-\U0001F77F"  # alchemical symbols
                               u"\U0001F780-\U0001F7FF"  # Geometric Shapes Extended
                               u"\U0001F800-\U0001F8FF"  # Supplemental Arrows-C
                               u"\U0001F900-\U0001F9FF"  # Supplemental Symbols and Pictographs
                               u"\U0001FA00-\U0001FA6F"  # Chess Symbols
                               u"\U0001FA70-\U0001FAFF"  # Symbols and Pictographs Extended-A
                               u"\U00002702-\U000027B0"  # Dingbats
                               u"\U000024C2-\U0001F251"
                               "]+", flags=re.UNICODE)
    return emoji_pattern.sub(r'', text)
```

The meticulous nature of these cleaning steps necessitated a systematic approach. Individual functions were created to execute each specific task. Subsequently, these functions were combined into a master cleaning function, which was then applied to the entire dataset. This holistic approach to text cleaning ensured that the tweets retained their core content while being relieved of extraneous information.

Fig. 8 Clean All function

```
def clean_all(text):
    text = remove_urls(text)
    text = remove_hashtag(text)
    text = remove_mention(text)
    text = remove_punctation(text)
    text = remove_numbers(text)
    text = remove_stop_words(text)
    text = remove_multiple_whitespace(text)
    text = remove_emojis(text)
    # remove space in beginning of text
    text = text.lower().strip()

    return text

df["Tweets"] = df["Tweets"].apply(clean_all)
```

3.2. Data processing

The data processing phase encompassed the determination of categories, tokenization, feature extraction, and the creation of a representative feature matrix and in the end training and evaulation of the model.

3.2.1. Determining categories

The initial step in this phase involved categorizing the tweets into distinct classes: symptoms, treatments, personal experiences, and scientific research. To achieve this, an iterative process was adopted. Each tweet was examined, and if it contained keywords indicative of a particular category, it was assigned to that category.

3.2.2. Tokenization and Frequency analysis

Tokenization, the process of breaking down textual content into individual words or tokens, was a crucial precursor to subsequent steps. To gain insights into the distribution of words within the dataset, a word frequency analysis was conducted. By calculating word frequencies, we could identify the most common and least common words used across all tweets. The nltk library was employed to compute word frequencies, with a focus on identifying patterns that might guide category assignment.

Fig. 9 Code for tokenization and frequency analysis

```
# Tokenization and frequency analysis
all_words = ' '.join(df['Tweets']).split()
word_counts = Counter(all_words)

from nltk.probability import FreqDist

# Calculate word frequencies
word_freq = FreqDist(all_words)

# Print the least common words and their frequencies
most_common_words = word_freq.most_common(1000)

print("Most common words and their frequencies:")
for word, count in most_common_words:
    print(f"{word}: {count}", end=', ')
```

3.2.3. Keyword-based Category Assignment

To accurately categorize tweets, predefined sets of keywords were created for each category. These keywords acted as indicators to match the content of the tweets with relevant categories. For instance, tweets containing words such as "symptoms," "fever," and "fatigue" were assigned to the symptoms category.

Fig. 10 Assigning words for each category

```
# Symptoms Keywords
symptoms_keywords = [
    "symptoms", "infection", "suffering", 'fever'
    "effects", "mild", "severe", "symptomatic",
    "chronic", "infections", "acute", "cough", "fatigue",
    "brain fog", "complications", "headaches", "breathlessness"
]

# Treatments Keywords
treatments_keywords = [
    "treatments", "vaccine", "prevent", "treatment", "vaccinated",
    "recovery", "therapy", "healthcare", "support", "medication",
    "boosters", "medical", "patients", "prevention"
]

# Personal Experiences Keywords
personal_keywords = [
    "i", "like", "me", "my", "im", "i'm", "i've", "experience", "feeling",
    "know", "think", "hope", "tried", "felt", "remember", "personally"
]

# Scientific Research Keywords
research_keywords = [
    "preprint", "characterizing", "scientific", "research", "study",
    "data", "evidence", "studies", "findings"
]
```

Fig. 11 Assigning categories for each tweet

```
categories = []

for i, tweet in enumerate(df['Tweets']):
    tweet_lower = tweet.lower()
    category = ''

    if any(keyword in tweet_lower for keyword in symptoms_keywords):
        category = 'symptoms'
    elif any(keyword in tweet_lower for keyword in treatments_keywords):
        category = 'treatments'
    elif any(keyword in tweet_lower for keyword in research_keywords):
        category = 'scientific research'
    else:
        category = 'personal experiences'

    categories.append(category)

df['Category'] = categories
```

3.2.4. Feature Extraction with Word2Vec

Feature extraction was a pivotal step to convert textual data into numerical vectors that could be used by machine learning algorithms. Word2Vec, a popular technique, was employed to create distributed representations of words. These embeddings capture the semantic meaning of words and phrases, thereby enhancing the quality of the feature vectors.

Fig. 12 Word2Vec

```
# Preprocessed text data as a list of sentences, where each sentence is a list of words
sentences = [tweet.split() for tweet in df_copy['Tweets']]

# Build a Word2Vec model
model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, sg=0)

# Function to calculate document embedding
def document_embedding(document, model):
    word_vectors = [model.wv[word] for word in document if word in model.wv]
    if not word_vectors:
        return None
    doc_embedding = sum(word_vectors) / len(word_vectors)
    return doc_embedding
```

3.2.5. *Sampling the Dataset*

Due to the large size of the dataset, a random sample of 20,000 rows was selected for further processing. This subsampling ensured computational feasibility while maintaining the representativeness of the data.

3.2.6. *Document Embedding*

For each tweet in the sampled dataset, document embeddings were generated using the Word2Vec model. This process transformed the tweet's content into a numerical vector, encapsulating its semantic meaning.

3.2.7. *Creating Feature Matrix and Labels*

The document embeddings were utilized to construct a feature matrix (X) and corresponding category labels (y). Each row in the matrix represented a tweet's content in vectorized form, while the labels indicated the respective categories.

3.2.8. *Training and Evaluation*

The sampled data was divided into training and testing sets to enable the training and evaluation of classification models. A Support Vector Classifier (SVC), a well-suited algorithm for text classification tasks, was trained on the training set and evaluated on the testing set. The evaluation involved calculating precision, recall, and F1-score, metrics that quantified the model's performance in terms of accuracy, completeness, and balance between precision and recall.

Fig. 13 Splitting dataset into train and test

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

3.3. *Results and discussion*

The culmination of the methodology and data processing efforts is reflected in the results of our text classification endeavor. The classification performance, as presented below, offers insights into the accuracy and effectiveness of the developed approach.

Fig. 14 Support Vector Classifier

```
# Train a Support Vector Machine (SVM) classifier
svm_classifier = SVC()
svm_classifier.fit(X_train, y_train)
```

Fig. 15 Model prediction

```
# Predict categories on the test set
y_pred = svm_classifier.predict(X_test)
```

3.3.1. *Classification Results*

The following table outlines the classification results obtained from the evaluation of our Support Vector Classifier (SVC) model. The metrics precision, recall, and F1-score provide a comprehensive view of the model's performance across the different categories.

	precision	recall	f1-score	support
personal experiences	0.79	0.92	0.85	1866
scientific research	0.96	0.91	0.94	830
symptoms	0.77	0.67	0.72	799
treatments	0.70	0.47	0.56	505
accuracy			0.81	4000
macro avg	0.81	0.74	0.77	4000
weighted avg	0.81	0.81	0.80	4000

Table 1 Evaluation scores

3.3.2. *Discussion*

The results exhibit a varying degree of success across the different categories. Notably, the "Personal Experiences" and "Scientific Research" categories exhibit higher precision and recall values, indicating a strong ability of the classifier to correctly identify and categorize these types of tweets. On the other hand, the "Symptoms" and "Treatments" categories show relatively lower precision and recall, suggesting that the classifier faces challenges in consistently identifying tweets belonging to these categories.

The overall accuracy of approximately 81% reflects the model's ability to correctly classify the majority of the tweets in the dataset. The macro average F1-score of around 0.77 indicates a satisfactory balance between precision and recall, with room for improvement in the "Symptoms" and "Treatments" categories.

The weighted average F1-score of approximately 0.80 further validates the model's effectiveness in handling imbalanced class distributions.

3.3.3. *Implications and Future Directions*

The achieved results underscore the potential of our classification approach to organize and categorize #LongCovid-related tweets. The insights extracted from this classification hold relevance for healthcare practitioners, researchers, and policymakers seeking to understand the landscape of discussions surrounding Long Covid.

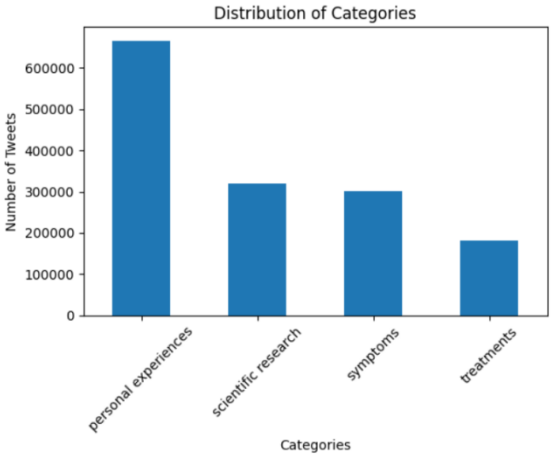
While the classification model showcases promising performance, several avenues for improvement exist. Fine-tuning the keyword-based category assignment, exploring advanced text embedding techniques, and employing ensemble models are potential strategies to enhance the model's precision and recall across all categories.

IV. DATA VISUALIZATION

Data visualization plays a pivotal role in conveying insights and patterns that might be concealed within the raw data. In this section, we present visualizations that provide a visual representation of the distribution of categories within the dataset and offer insights into the characteristics of the sampled data.

To understand the distribution of categories within the entire dataset, a bar plot was generated showcasing the number of tweets corresponding to each category. The following figure illustrates this distribution:

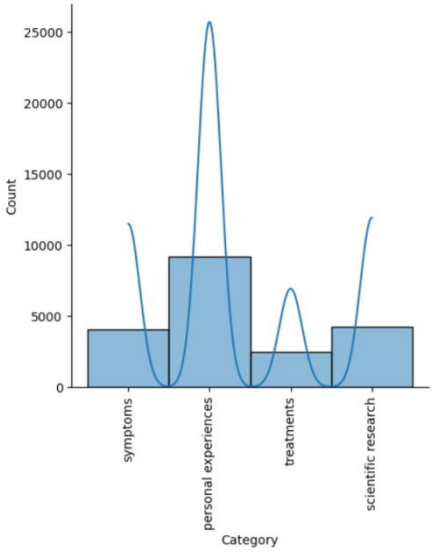
Fig. 16 Bar plot of the distribution of the categories



From the bar plot it can be seen that the ‘personal experiences’ category prevails, then ‘scientific research’ and ‘symptoms’ follow with similar numbers and the least number of tweets are in the ‘treatments’ category.

In addition to examining the distribution of categories within the entire dataset, we also explored the category distribution within the sampled data. The following density plot illustrates this distribution:

Fig. 17 Bar plot of category distribution within the sampled data



This density plot provides insights into the distribution of categories within the sampled data, enabling us to assess whether the sampling process effectively captured the original distribution.

V. CONCLUSION

This paper embarked on the journey of classifying #LongCovid-related tweets into distinct categories, aiming to facilitate the extraction of meaningful insights from the collective discourse.

Through a multi-faceted approach that involved data collection, preprocessing, text classification, and evaluation, this paper demonstrated the potential of text classification techniques to categorize diverse tweets. The results obtained showcased a commendable accuracy in classifying "Personal Experiences" and "Scientific Research," while indicating areas for improvement in the "Symptoms" and "Treatments" categories.

The methodology presented underscores the importance of robust data preparation, encompassing handling missing values, text cleaning, and keyword-based category assignment. By leveraging Google Colaboratory and Python, a comprehensive implementation was achieved,

showcasing the feasibility of this approach even with large-scale datasets.

The classification results offer insights into the distribution and characteristics of #LongCovid-related tweets, providing valuable information for healthcare practitioners, researchers, and policymakers. The visualizations further complement these insights, offering a tangible representation of category distributions.

As a holistic endeavor, this paper bridges the gap between the massive volume of unstructured text data and meaningful insights. While the classification model exhibited promising performance, there remains room for refinement and enhancement. Future research could explore advanced techniques in text embedding, feature engineering, and ensemble models to improve classification accuracy.

In conclusion, the classification of #LongCovid-related tweets stands as a testament to the potential of text classification techniques in organizing and deciphering complex textual data. Through a combination of rigorous methodology, advanced tools, and thoughtful evaluation, this endeavor contributes to the understanding of Long Covid and offers a framework to navigate the sea of digital discourse for actionable insights.

REFERENCES

[1] Sklearn. (2021). Support Vector Machines (SVM). scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org/stable/modules/svm.html

[2] 'https://github.com/atodorovska/VNP-Exercises'

[3] 'https://www.kaggle.com/code/matt0922/tweets-processing'

[4] 'https://www.kaggle.com/code/peacechoy/tweets-processing-ec9722'