Social Media Analysis: Designing, Describing, and Comparing Neural Network Models for Emotion Detection in Tweets.

Student ID: 23220052

I - Introduction:

Emotions are complex psychological states that are fundamental to human behaviour, influencing how we perceive and interact with the world around us [1]. Indeed, these emotions consist of the individual feeling of an experience and vary significantly from one person to another [2]. Their significance triggers physiological changes in bodies such as in heart rate, hormone level and breathing, as well as conscious or unconscious outward expressions such as body language, vocal notes and facial expressions [3]. However, emotions come as basic and complex. Indeed, basic emotions are universal and experienced by all humankind regardless of their culture and background [1]. According to psychologist Paul Ekman, these include happiness, sadness, fear, anger, surprise, and disgust [4]. In contrast, complex emotions are a combination of the above, influenced by personal experience and social context such as jealousy, shame and pride [1]. These emotions are essential for social interaction as they facilitate communication and enhance our ability to understand and respond to people's emotions, improving empathy and cooperation [5]. Moreover, emotions also influence our decision-making, often guiding us toward our perceived optimal choice [6]. Therefore, understanding and managing emotions is essential for maintaining mental health and building resilience against stress and adversity. Indeed, it is crucial to detect emotions. This process involves identifying and categorizing emotional tones expressed within textual data, images and audio using Natural Language Processing (NLP) [7]. Analysing this data helps understand human sentiment and behaviours, as emotions can be conveyed explicitly through words or implicitly through context and word combinations [8]. One of the most frequent areas associated with the applications of emotion detection is social media analysis. Indeed, every day, platforms such as Twitter, Facebook, and Instagram receive a huge traffic and content generated by users [9]. Therefore, analyzing this content for emotional tone offers valuable insights into social opinion, brand perception, and trends [10].

Neural Networks (NNs) in computer science are information models imitating the anatomy and functioning of the human brain [11]. These are made of interconnected layers of nodes in which inputs are processed into an output [11]. Indeed, they use labelled data to train algorithms to learn complex patterns and relationships and classify or predict results from new input data [12]. Some examples include Convolutional Neural Networks (CNNs) - typically used for image and video recognition tasks, but also effective for text categorization tasks [13]; and Recurrent Neural Networks (RNNs) - used in time series, speech recognition, and text analysis tasks [14]. Due to their ability to capture intricate patterns in textual data, such models are able to

accurately determine emotive content by identifying key phrases indicative of emotions and understanding context and the sequential nature of language [8].

⇒ In this report, we will develop and compare different NN models for emotion detection, explain the research methodology that has been followed, present and discuss our results and finally, address some of the ethical concerns of our study. By evaluating the performance of these models, we aim to determine the most effective approach for emotion detection in textual data, specifically in the context of social media analysis.

II - Literature Review:

Emotion detection in text has become a crucial aspect of sentiment analysis, due to its applications in social media analysis [8]. This has led to various research focusing on its detection and prediction through NLP and ML algorithms. Indeed, NN-based approaches seem to contribute significantly to emotion detection in text.

Being a subcategory of ML models, NNs greatly differ from classic ML in terms of architecture, complexity, and application [15]. Indeed, NNs consist of layers of interconnected nodes (neurons) allowing them to automatically learn and extract features from high-dimensional data like images, audio, and text [11]. Despite being more complex, less interpretable and requiring significant computational power to train compared to the simpler architectures of classic ML, NNs are highly flexible and perform extremely well on large amounts of labelled data and complex tasks, while classical ML perform better on small datasets and might struggle with very high-dimensional data [16]. Indeed, according to Machová et al., (2023) study, NN models outperform classic ML models with an accuracy of 91.0% and excel in multi-class classification tasks involving six emotions, achieving an F1-score of 0.95 for sadness [17]. In terms of CNN models, these classes of DL models use convolutional layers with filters that slide over the input to capture spatial hierarchies of features, making them particularly effective for text-based emotion detection [18]. Indeed, Cahyani et al., (2022) study combined CNN and Bidirectional Long Short-Term Memory (BiLSTM) networks to detect emotions in text, utilizing Word2Vec and GloVe embeddings [19]. This consisted firstly of distinguishing emotion from no emotion and then classifying five emotion types (happiness, anger, sadness, fear, surprise). As a result, the Word2Vec-CNN-BiLSTM model achieved slightly higher results compared to the GloVe-CNN-BiLSTM model, with accuracy values of 84.34% and 83.88% respectively, as well as better overall precision, recall and F1-score [19]. On the other side, Bharti et al., (2022) study used CNN integrated with Bidirectional Gated Recurrent Units (Bi-GRU) to classify emotions within sentences, tweets, and dialogues, achieving an accuracy of 79.46% compared to 78.97% for the SVM model [20]. However, when comparing CNN-Bi-GRU model to a simple CNN model, the authors pointed out that the CNN-Bi-GRU had yielded a slightly higher accuracy (79.46% compared to 79.32%), but lower precision, recall and F1-score values (80.62%, 79.64%, 80.09% compared to 82.12%, 79.92% and 80.76%) [20]. In contrast to CNNs, RNNs are designed for sequential data, making them effective for NLP usage, where the order of data points is crucial [14]. They process inputs in sequence, retaining information through their

internal state, which helps capture temporal dependencies [14]. While CNNs generally train faster and can achieve high accuracy, RNNs excel in tasks requiring context or sequence understanding but often require longer training times and more data due to their complexity [21]. Indeed, Yohanes et al., (2023) study, which compared LSTM, BiLSTM, and GRU RNN models using the ISEAR dataset, highlighted that RNN GRU achieved the highest accuracy (60.26%) as well as recall, precision and F1 score, compared to BiLSTM (59.3%), and LSTM (57.65%) [22]. Additionally, it emphasized the model's simplicity and effectiveness in emotion detection, while noting potential improvements with more robust datasets and preprocessing.

III - Methodology:

A. Our dataset:

The “emotion_sentiment_dataset.csv” used for our NN model development was retrieved from Kaggle [23]. It consists of a collection of 839,555 English tweets annotated with 13 emotions: love, hate, neutral, surprise, anger, relief, happiness, enthusiasm, sadness, empty, fun, boredom and worry. Indeed, each entry consists of a text associated to its corresponding label, which indicates the predominant emotion conveyed. This way, through identifying and classifying emotions expressed in tweets, we will be able to detect emotion on social media and contribute to a broader understanding of emotional expression in written content. Therefore, our project aims to design, train, test and deploy our NN models on this labelled dataset with known outcome, to accurately detect emotion.

B. Data Exploration and Pre-processing:

Before starting our project, we first import all required libraries for our program solution.

In order to detect emotions, we first import and load our dataset from the “emotion_sentiment_dataset.csv” file. Then, we thought it would be beneficial to evaluate our dataset number of rows to have an idea of our dataset shape. Given its large number, we have decided to reduce our dataset size for further processing while maintaining reproducibility by randomly sampling it to 100,000 rows to avoid algorithm intensity and reduce processing time. The sampled dataframe was saved to a new CSV file named 'sampled_tweet_emotions_df.csv', excluding the index column, as it did not add any value to our analysis (Figure 1). Once done, we examined the unique emotion values of our dataset, allowing us to understand and validate the different emotion categories represented in our sampled dataset. Subsequently, we inspected our dataset to gain insights into its missing values, duplicated rows and emotion distribution. Such insights allow us to understand the overall behaviour and spread of our dataset, guiding our selection of appropriate analysis techniques and models. Indeed, as we clearly noticed a data imbalance, specifically for the “neutral” emotion that may impact the performance of any emotion analysis models trained on this dataset, we decided to address it by downsampling our 'neutral' class to a minority class mean count in order to reduce our its size, avoid dominance in the dataset and improve model accuracy and fairness (Figure 2).

Given that our input text consists of social media messages, we thought it would be interesting to create a dictionary of common chat words that are likely to appear in tweets. Indeed, we mapped and replaced common chat abbreviations and acronyms with their full meanings to enhance text processing and understanding in applications such as social media analysis (Figure 3). Once done, our dataset underwent pre-processing steps to ensure uniformity and compatibility for NN models. Therefore, to ensure that our text is preprocessed effectively, we first removed any URLs from our text as they do not contribute to any semantic meaning. Next, we converted our text to lowercase to ensure uniformity and matching words accurately as well as prevent case sensitivity issues. After that, we removed any punctuation, symbols and numbers as they do not carry semantic meaning on their own, to ensure that we have clean tokens and to prevent symbols from being treated as separate tokens. Indeed, this is done to simplify our text, reduce the complexity of the vocabulary, avoid unnecessary tokens and improve model performance by focusing on textual contents. Following this, we tokenize our text into words, remove stop words, making our text prepared for stemming and lemmatization which consists of reducing the words into their base forms to ensure that they are in their standard, normalized form, which is crucial for consistency and analysis (Figure 4). After that, we decided to explore and visualize the top 5 words for each unique emotion, to help us understand the vocabulary commonly linked to each emotion, hence providing insights into the specific terms used in the context of different emotions. Then, given that our emotion column consists of non-numerical data, we convert them into numerical equivalents, by mapping our dataset emotions to ensure consistent handling of our data (Figure 5). Once our dataset is pre-processed and cleaned, we split it into training (70%), testing (25%) and deployment (5%) sets which we will use to train, evaluate and validate our model. However, prior to developing our model, we identified and visualized emotion value counts for each dataset. This step is crucial as it ensures that NN models perform well across all classes, avoiding bias towards the majority class and thereby providing more accurate and reliable predictions. Finally, we performed sequence padding for each dataset to ensure they are of uniform length before defining the input size for the embedding layer (Figure 6). This is essential for initializing the embedding layer in our NN models, as it defines the vocabulary size that the layer will use to map token indices to dense vectors.

```
#We first ensure reproducibility by setting a random seed
random_seed = 42

#Then we sample our dataset to 100,000 rows
sampled_tweet_emotions_df = tweet_emotions_df.sample(n=100000, random_state=random_seed)

#Finally we print the number of rows in our sampled dataset
num_sampled_rows = sampled_tweet_emotions_df.shape[0]
print(f"Number of rows in the sampled dataset: {num_sampled_rows}")

#And save it to a new CSV file
sampled_tweet_emotions_df.to_csv('sampled_tweet_emotions_df.csv', index=False)
```

Figure 1: *Code snippet of data sampling to 100,000*

```python
#We first separate majority and minority classes
df_majority = sampled_tweet_emotions_df[sampled_tweet_emotions_df.Emotion == 'neutral']
df_minority = sampled_tweet_emotions_df[sampled_tweet_emotions_df.Emotion != 'neutral']

#We then determine the mean number of samples for the minority class
mean_count = int(df_minority['Emotion'].value_counts().mean())

#And under-sample the majority class to match the minority class count
df_majority_downsampled = resample(df_majority,
                                   replace=False,
                                   n_samples=mean_count,
                                   random_state=42)

#Then, we combine minority class with downsampled majority class
sampled_tweet_emotions_balanced_df = pd.concat([df_majority_downsampled, df_minority])

#We shuffle our dataset to mix the classes
sampled_tweet_emotions_balanced_df = sampled_tweet_emotions_balanced_df.sample(frac=1, random_state=42).reset_index(drop=True)

#And save the balanced dataframe to our original dataframe
sampled_tweet_emotions_df = sampled_tweet_emotions_balanced_df

#And check our balanced class distribution
balanced_emotion_counts = sampled_tweet_emotions_df['Emotion'].value_counts()
print("Balanced Emotion Value Counts:")
print(balanced_emotion_counts)
```

Figure 2: *Code snippet showing the downsampling of our dataset*

```python
#We create a copy of our text column
text_series = sampled_tweet_emotions_df['text']

#Then, we replace chat words with their full forms
for chat_word, full_form in chat_words.items():
    text_series = text_series.str.replace(r'\b' + chat_word + r'\b', full_form, case=False)

#And assign the modified series back to our dataframe
sampled_tweet_emotions_df['text'] = text_series

#And finally print our updated dataframe
print(sampled_tweet_emotions_df.head())
```

Figure 3: *Code snippet illustrating the replacement of common chat abbreviations and acronyms to their full meanings.*

```python
stop_words = set(stopwords.words('english'))
stemmer = PorterStemmer()
lemmatizer = WordNetLemmatizer()

#Then we process each text entry in our dataframe
for index, row in sampled_tweet_emotions_df.iterrows():
    text = row['text']

    #We first remove URL
    text = re.sub(r'\[.*?\]', '', text)

    #Then, we convert text to lowercase
    text = text.lower()

    #And remove non-alphabetic characters
    text = re.sub(r'[^a-zA-Z\s]', '', text)

    #And numbers
    text = re.sub(r'\d+', '', text)

    #Then, we tokenize our text into words
    tokens = word_tokenize(text)

    #Remove stopwords
    tokens = [word for word in tokens if word not in stop_words]

    #After that, we stem words
    stemmed_tokens = [stemmer.stem(word) for word in tokens]

    #And lemmatize them
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in stemmed_tokens]

    #Finally, we join tokens back into a single string
    clean_text = ' '.join(lemmatized_tokens)
```

Figure 4: *Code snippet showing data pre-processing steps*

```python
#We first define the emotion mapping
emotion_mapping = {
    'neutral': 0, 'love': 1, 'happiness': 2, 'sadness': 3,
    'relief': 4, 'hate': 5, 'anger': 6, 'fun': 7,
    'enthusiasm': 8, 'surprise': 9, 'empty': 10, 'worry': 11, 'boredom': 12
}

#And then convert our "Emotion" column to numerical values using the above mapping
sampled_tweet_emotions_df['Emotion'] = sampled_tweet_emotions_df['Emotion'].map(emotion_mapping)
print(sampled_tweet_emotions_df.head())
```

Figure 5: *Code snippet illustrating emotion mapping of our dataset*

```python
#Then, we pad sequences
X_train_padded = pad_sequences(X_train_sequences, maxlen=maxlen_train)
X_test_padded = pad_sequences(X_test_sequences, maxlen=maxlen_train)
X_deploy_padded = pad_sequences(X_deploy_sequences, maxlen=maxlen_train)

#And check their shapes before printing them
print(f"Padded Training data shape: {X_train_padded.shape}")
print(f"Padded Training data: {X_train_padded}")

print(f"\nPadded Testing data shape: {X_test_padded.shape}")
print(f"Padded Testing data: {X_test_padded}")

print(f"\nPadded Deployment data shape: {X_deploy_padded.shape}")
print(f"Padded Deployment data: {X_deploy_padded}")
```

Figure 6: *Code snippet performing sequence padding for each dataset*

C. Classifiers Training, Evaluation, and Comparison:

Based on our literature review, we decided to select RNN-GRU and CNN-GRU models, trained, tested and deployed them to evaluate their accuracies and determine the most suitable one for emotion detection. We began by defining and configuring our RNN-GRU model (Figure 7) by adding an Embedding layer, to transform integer sequences into dense vectors of a fixed size (100-dimensional in this case); a Bidirectional layer that wraps a GRU layer with 128 units, enabling the model to learn dependencies in both forward and backward directions within the sequence; a BatchNormalization layer to normalize activations and improve training stability followed by a Dropout layer with a dropout rate of 0.5 to prevent overfitting by randomly setting half of the input units to zero during training. Then, our model includes a Dense layer with 64 units and ReLU activation, followed by another Dropout layer with a 0.5 dropout rate and an output Dense layer with 13 units and softmax activation is added to classify the data into 13 categories. After that, our model was compiled using the Adam optimizer, sparse categorical cross-entropy loss function, an accuracy metric, and a model_rnn.summary() function was called to display our model's architecture, including the number and types of layers, output shapes, and the total number of parameters. Then we trained our defined RNN model for 20 epochs with a batch size of 32 with an EarlyStopping set with a patience parameter of 3, which means that training will be halted if there is no improvement in the validation loss for 3

consecutive epochs. Following that, we evaluated our trained RNN model's performance on our testing data, generating a classification report to provide a detailed breakdown of precision, recall, and F1-score for each class, as well as a confusion matrix to compare the true labels with the predicted labels. Then, our trained model was saved to “rnn_model.h5” file, allowing easy storage for later retrieval. Finally, we deploy our model on our deployment data and provide a classification report and confusion matrix to visualize our RNN model performance by comparing predicted versus true labels, before saving our dataframe to “deployment_predictions_with_text_rnn.csv” file and visualise our RNN model training, testing and deployment accuracies.

```
#We first define our model
model_rnn = Sequential()

#Them, add an embedding layer
model_rnn.add(Embedding(input_dim=input_size, output_dim=100,input_shape=(79,)))

#As well as a bidirectional-GRU layer
model_rnn.add(Bidirectional(GRU(128)))

#A batch normalization layer
model_rnn.add(BatchNormalization())

#A dropout regularization
model_rnn.add(Dropout(0.5))

#A dense layer with 64 units and ReLU activation
model_rnn.add(Dense(64, activation='relu'))

#Another dropout regularization
model_rnn.add(Dropout(0.5))

#and an output layer with 6 units for 6 labels and softmax activation
model_rnn.add(Dense(13, activation='softmax'))

#We then compile it
model_rnn.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

#And display our model summary
model_rnn.summary()
```

Figure 7: *Code snippet defining our RNN model*

```
#We first define our CNN-GRU model
model_cnn = Sequential()

#Add an embedding layer
model_cnn.add(Embedding(input_dim=input_size, output_dim=100, input_length=79))

#As well as a convolutional layer with 128 filters, kernel size of 5, and ReLU activation
model_cnn.add(Conv1D(filters=128, kernel_size=5, activation='relu'))

#A max pooling layer
model_cnn.add(MaxPooling1D(pool_size=2))

#A batch normalization layer
model_cnn.add(BatchNormalization())

#Then, we flatten the output to feed into fully connected layers
model_cnn.add(Bidirectional(GRU(128)))

#Add dropout regularization
model_cnn.add(Dropout(0.5))

#And a dense layer with 64 units and ReLU activation
model_cnn.add(Dense(64, activation='relu'))

#A dropout regularization
model_cnn.add(Dropout(0.5))

#As well as the output layer with 13 units for 13 labels and softmax activation
model_cnn.add(Dense(13, activation='softmax'))

#Before compiling it
model_cnn.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

Figure 8: *Code snipper defining our CNN model*

In terms of our CNN-GRU model, we first defined and configurated it (Figure 8) by adding an embedding layer; a 1D convolutional layer with 128 filters and ReLU activation which captures local patterns in the text; a max pooling layer to reduce the dimensionality; a batch normalization layer to normalize the outputs for stability; a bidirectional GRU layer with 128 units to capture sequence dependencies from both directions; dropout layers, each with a rate of 0.5, are added before and after a dense layer with 64 units and ReLU activation to prevent overfitting; and finally an output layer that has 13 units with softmax activation to produce a probability distribution over the classes is added. The model was then compiled using the Adam optimizer, sparse categorical cross-entropy loss, and accuracy as the evaluation metric, creating a robust architecture for text classification. Then we used similar steps for the training, testing and deployment of our RNN-GRU model. Finally, we decided to generate a bar plot showing both our RNN-GRU and CNN-GRU training, testing and deployment accuracies for a clear comparison of our models.

IV - Results:

Initially, our dataset was complete and had no missing values, allowing straightforward use. However, it was clearly imbalanced, specifically the majority class “neutral” which consisted of 80,396 cases (Figure 9), compared to a mean of minority classes of 1633 cases; while “love” and “boredom” (minority class) consisted of 4804 and 14, respectively. Therefore, when downsampling the neutral class for a more balanced dataset, its row number dropped from 100,000 to 21,237. Moreover, when exploring and visualizing the top 5 words for each unique emotion (Figure 10), we noticed that the top one most common word was “feel”, followed by the emotion adjectives, and their synonyms. For instance, the top words associated with “happiness” are “feel” with 3449 iteration, “happi” with 1244, “enjoy” with 685 and “content” with 682; while the top words associated with “worry” are “feel” with 600 iteration, “worri” with 248, “concern” with 165 and “anxieti” with 154. Upon splitting our dataset, our training data consisted of 14865 cases, our testing dataset of 5310 cases, and our deployment data of 1062 cases. We then identified and visualized emotion value counts for each dataset (Figure 11), and noticed that datasets were more balanced, with the majority class being “love” throughout the training (3365), testing (1197) and deployment dataset (242); while the minority class being “boredom” with 10, 3 and 1 instances respectively. This correlates with the 75%, 25% and 5% ratio of our previous findings. Then, when sequence padding for each split dataset, we noticed that the maximum sequence length for training was 40, which was the highest compared to training (31) and deployment (33). Therefore, we reshape our padded training data shape to (14865, 40), training one to (5310, 40) and deployment one to (1062, 40). As we prepared our datasets for defining the NN model, we found an input size for our embedding layer of 10227. Indeed, we first defined our RNN model, which yielded 88.96% accuracy on training dataset, 85.25% on testing dataset and 84.93% on deployment dataset (Figure 12). Our performance of our model on testing dataset generated an accuracy of 85%, with high precision and recall for classes 3, 5, 9, and 10, a poor performance for class 12 with precision, recall, and F1-Score all at 0.00, likely due to very low support (only 3 instances), and disparities within class 2 (high precision 0.98/low recall 0.31 and F1-score 0.47) and class 7 (high precision 0.97/low recall 0.63 and F1-score

0.77) (Figure 13). Concerning the classification report for our deployment dataset, it represented a high accuracy of 85%, with high precision and recall for classes 3, 5, 6, 9, and 10, a poor performance for class 12 with precision, recall, and F1-Score all at 0.00, probably due to very low support (only 1 instance), and disparities within class 2 (high precision of 0.98/low recall 0,88) and class 11 (low precision 0.33 and F1-score 0.49/high recall 0.95) (Figure 14). Then, we defined our CNN model, which yielded 99.43% accuracy on training dataset, 96.50% on testing dataset and 97.18% on deployment dataset (Figure 12). Our classification report for testing dataset showed an accuracy of 96%, with a high precision and recall for classes 1, 2, 3, 4, 5, 6, 8, 9, 10 and 11, a poor performance for class 12 with precision, recall, and F1-Score all at 0.00, likely due to very low support (only 3 instances), and disparities within class 0 (low precision 0.89/high recall 0.96 and F1-score 0.93) and class 7 (high precision 0.92 and F1-score 0.91/low recall 0.89) (Figure 15). In terms of classification report for our deployment dataset, it yielded a high accuracy of 97%, with high precision, recall and F1-score for all classes, though a poor performance for the 12th class with precision, recall, and F1-Score all at 0.00, likely due to very low support (only 1 instance), and perfect predictions for class 9 (with 1.00 through precision, recall and F1-score) (Figure 16).

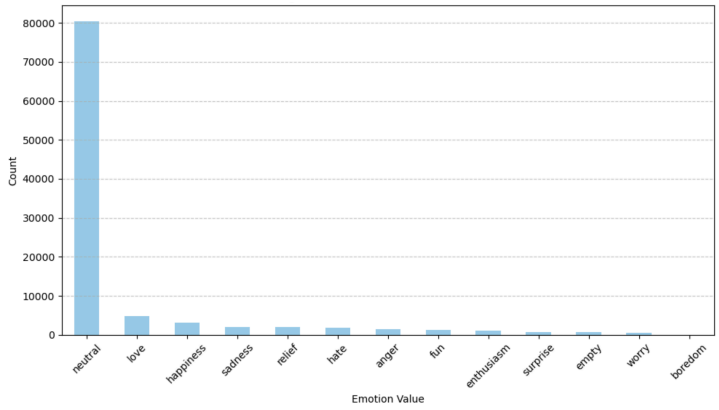

Figure 9: *Emotion value distribution*

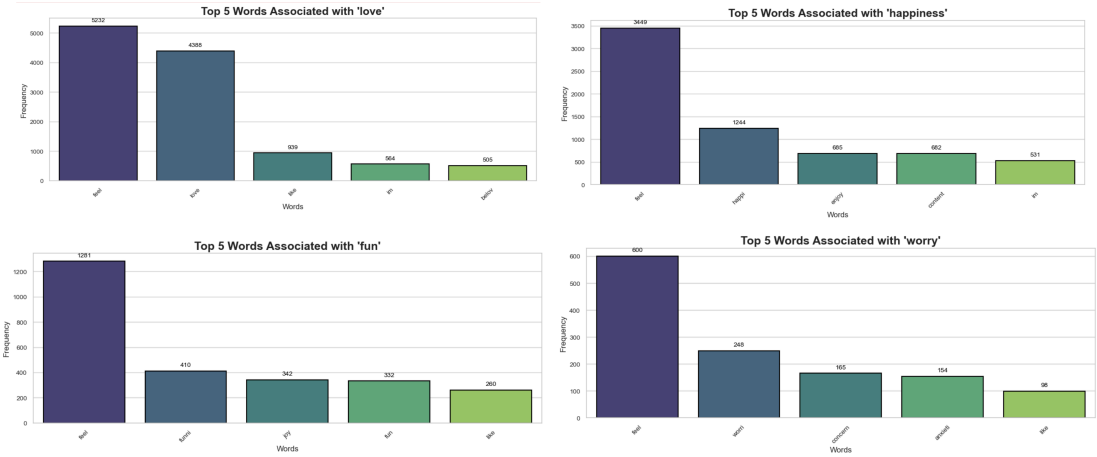

Figure 10: *Visualisation of the top 5 words for "love", "happiness", "fun" and "worry" emotions*

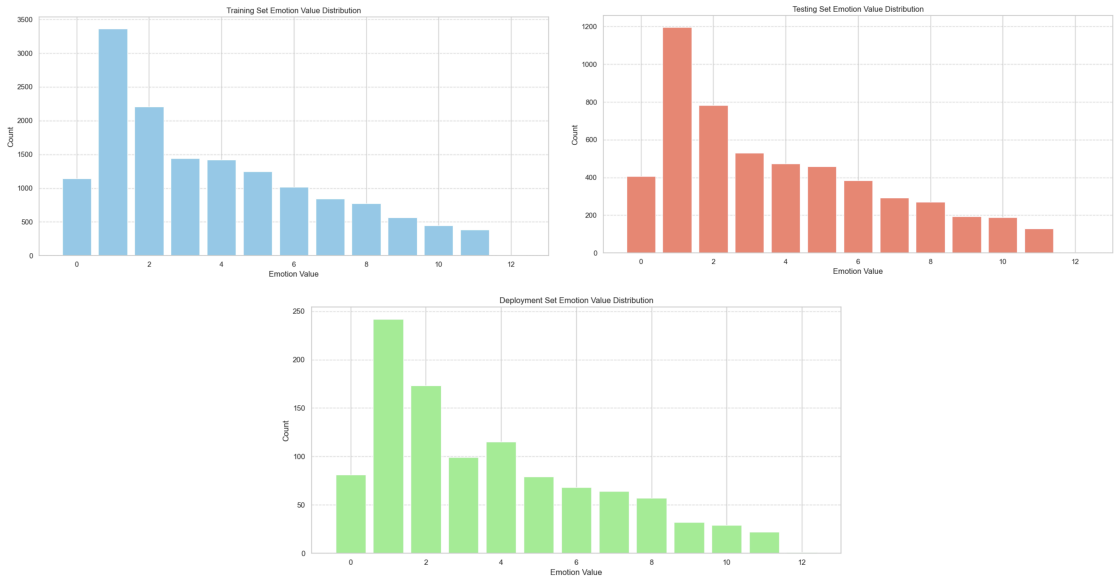

Figure 11: *Visualisation of emotion value counts for training, testing and deployment datasets*

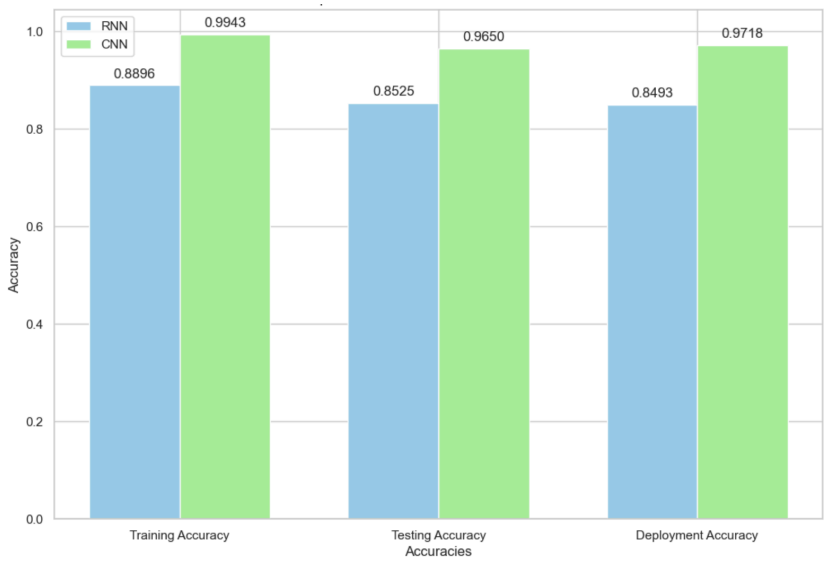

Figure 12: *Training, testing and deployment accuracies for both our RNN and CNN models*

	precision	recall	f1-score	support
0	0.87	0.91	0.89	406
1	0.79	0.99	0.88	1197
2	0.98	0.31	0.47	782
3	0.99	0.97	0.98	531
4	0.88	0.98	0.93	474
5	0.99	0.98	0.98	458
6	0.95	0.94	0.95	385
7	0.97	0.63	0.77	293
8	0.69	0.99	0.81	270
9	0.99	0.96	0.98	193
10	0.99	0.94	0.97	188
11	0.41	0.98	0.58	130
12	0.00	0.00	0.00	3
accuracy			0.85	5310
macro avg	0.81	0.81	0.78	5310
weighted avg	0.89	0.85	0.84	5310

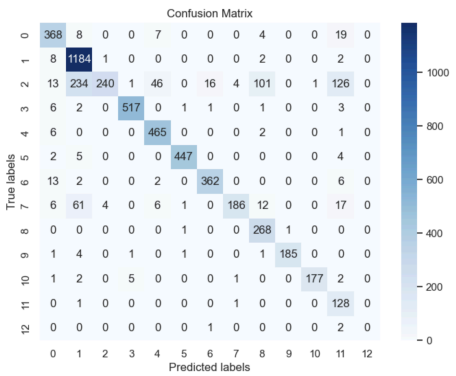

Figure 13: *Classification report and confusion matrix on our testing dataset using our RNN model*

	precision	recall	f1-score	support
0	0.93	0.91	0.92	81
1	0.80	0.99	0.88	242
2	0.98	0.31	0.47	173
3	0.98	1.00	0.99	99
4	0.93	0.98	0.95	115
5	0.99	1.00	0.99	79
6	0.99	0.99	0.99	68
7	1.00	0.66	0.79	64
8	0.60	0.98	0.75	57
9	1.00	0.94	0.97	32
10	1.00	0.93	0.96	29
11	0.33	0.95	0.49	22
12	0.00	0.00	0.00	1
accuracy			0.85	1062
macro avg	0.81	0.82	0.78	1062
weighted avg	0.90	0.85	0.83	1062

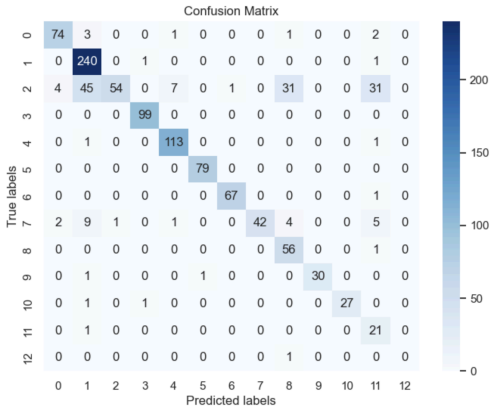

Figure 14: *Classification report and confusion matrix on our deployment dataset using our RNN model*

	precision	recall	f1-score	support
0	0.89	0.96	0.93	406
1	0.99	0.96	0.98	1197
2	0.95	0.96	0.96	782
3	0.99	0.95	0.97	531
4	0.98	0.98	0.98	474
5	0.95	0.99	0.97	458
6	0.98	0.99	0.99	385
7	0.92	0.89	0.91	293
8	0.96	0.99	0.97	270
9	0.99	0.98	0.99	193
10	0.98	0.97	0.98	188
11	0.95	0.95	0.95	130
12	0.00	0.00	0.00	3
accuracy			0.96	5310
macro avg	0.89	0.89	0.89	5310
weighted avg	0.97	0.96	0.96	5310

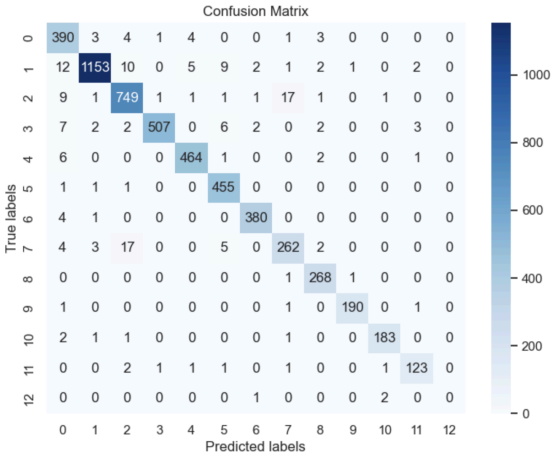

Figure 15: *Classification report and confusion matrix on our testing dataset using our CNN model*

	precision	recall	f1-score	support
0	0.92	0.98	0.95	81
1	0.98	0.96	0.97	242
2	0.97	0.95	0.96	173
3	0.99	0.99	0.99	99
4	0.99	0.98	0.99	115
5	0.92	1.00	0.96	79
6	1.00	0.99	0.99	68
7	0.98	0.94	0.96	64
8	0.97	1.00	0.98	57
9	1.00	1.00	1.00	32
10	0.97	0.97	0.97	29
11	0.96	1.00	0.98	22
12	0.00	0.00	0.00	1
accuracy			0.97	1062
macro avg	0.90	0.90	0.90	1062
weighted avg	0.97	0.97	0.97	1062

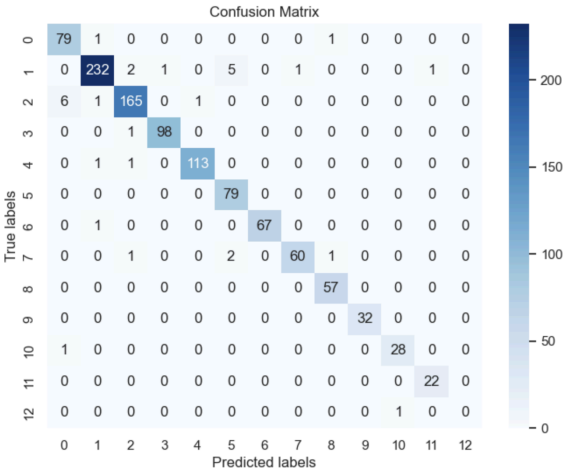

Figure 16: *Classification report and confusion matrix on our deployment dataset using our CNN*

V - Discussion:

A. Discussion of results:

The use of NN methods in our emotion detection study resulted in promising outcomes. Initially, our dataset was large with 839,555 input text, did not contain any missing values or duplicated rows, but had a clear imbalance for its “neutral” class. Therefore, it was sampled to 100,000 to improve computational efficiency and manageability. Indeed, this allowed faster model training and testing, making it feasible to conduct multiple experiments and optimize hyperparameters within reasonable resource limits. However, the sampling exposed it to risks such as potential loss of important information and exacerbation of class imbalance issues, which could affect model generalization and introduce biases [23]. Therefore, it is crucial to ensure the sampling process is random and representative to maintain class distribution and dataset integrity and to develop and evaluate robust models [24]. Nevertheless, our dataset did not have missing values or duplication rows making it. Moreover, the “neutral” class was downsampled to the mean average of the other classes. Please note that we first thought about downsampling it to 5000, however, given that our other classes' mean was lower, it could have resulted in an excessive loss of information and there could still lead to a significant imbalance potentially causing the majority class to dominate [25]. Therefore, by downsampling our majority class based on other classes' averages, we provided a more uniform distribution, making it more proportional and not excessively reducing data. However, by calculating the mean average of minority classes, a layer of complexity is added and there could still be some loss of data [25]. Please note that since this step was done randomly, every time we run the code, it provides us with different accuracies, as the sampled dataset is modified. Moreover, our training, testing and deployment datasets exhibited a similar emotion distribution, indicating a maintained balance of classes across them. This ensures that each subset of the data represents the overall distribution of emotions, leading to more reliable and generalizable model performance [26]. Indeed, by maintaining consistent class distribution, the model is less likely to be biased towards certain emotions, thus improving the validity and accuracy of its predictions during training, testing, and real-world deployment [27]. Finally, when comparing both our RNN and CNN models for emotion detection, it is evident that our CNN model outperforms the RNN model across multiple metrics. Indeed, it achieved significantly higher accuracy rates (99.43% training, 96.50% testing, and 97.18% deployment) compared to the RNN (88.96% training, 85.25% testing, and 84.93% deployment). This might be due to the CNN's ability to better handle large datasets through convolutional layers and its consistent high performance, even in real-world scenarios, underscores its robustness [18]. However, while RNNs are advantageous for sequential data due to their ability to capture temporal dependencies [8], their lower recall rates indicate a tendency to miss true positive cases. Indeed it showed notable disparities in performance, particularly struggling with classes with low support, such as class 12. In contrast, our CNN demonstrated robust precision and recall across a broader range of classes, and near-perfect predictions, particularly for class 9, which achieved perfect precision, recall, and F1-score, hence showcasing its strong generalization capabilities (refer to Part II). Finally, both models share a common weakness in dealing with rare classes. Indeed, our RNN model performance was hindered by its lower recall, suggesting it misses a higher number of true positive cases,

hence highlighting its inconsistencies and limitations in dealing with imbalanced datasets. Although these disparities were less pronounced in our CNN model suggests areas for improvements. Indeed, to enhance our models for emotion detection, several strategies can be recommended. Firstly, addressing the class imbalance is crucial. This can be done through SMOTE techniques or data augmentation methods to create synthetic examples of underrepresented classes, thereby improving the models' ability to detect rare emotions [28]. Secondly, leveraging hybrid models that combine the strengths of RNNs and CNNs might capture both temporal dependencies and spatial features more effectively, potentially leading to better performance across all classes [29]. Thirdly, advanced techniques like transfer learning or more complex architectures could further enhance the model's ability to generalize [30]. Additionally, fine-tuning hyperparameters through methods like grid search or Bayesian optimization could further optimize model performance [31]. Moreover, incorporating attention mechanisms in RNNs could help the model focus on more relevant parts of the sequences, thereby improving recall rates [32]. Finally, continuous monitoring and improvement of the models with updated data will ensure their robustness and accuracy over time, helping maintain high accuracy and generalization capabilities in real-world applications [33].

B. Ethical considerations:

There are ethical considerations to address when detecting emotion on social media platforms.

First of all, given that tweets contain sensitive information and personal expressions privacy concerns are paramount. Indeed, using this data without explicit consent can have an impact on individuals' privacy rights [34]. Therefore, as per our dataset, it is crucial to anonymize data and obtain necessary permissions to protect users' identities [35]. For example, Zimmer and Proferes (2014) paper highlights the ethical implications of using Twitter data for research without users' knowledge, emphasizing the need for transparency and consent to maintain ethical standards [36]. Moreover, emotion detection models could be employed for malicious intents, such as targeted advertising or political propaganda, which can influence public opinion and behaviour in unethical ways [37]. For example, the recent Cambridge Analytica scandal demonstrated how data obtained from social media networks could be used to manipulate voters’ choices during elections, thus sparking concerns about privacy and ethical usage of data [38]. Furthermore, models trained on imbalanced datasets may result in the under-representation of some emotions or minority groups, thereby leading to skewed results that perpetuate existing biases [39]. This could result in unfair treatment or misinterpretation of specific user groups' emotions. Additionally, the highly subjective nature of emotions and context-dependency of emotional expression may not be interpreted similarly by different people or in different contexts, hence questioning the validity of emotion detection models [40]. Indeed, while detecting emotions such as worry, sadness, or relief can be beneficial for monitoring well-being, it also requires careful handling to avoid stigmatization or unwarranted interventions [41]. For example, a study by Mohammad and Turney (2013) found that sentiment analysis systems often struggle with correctly understanding the context of emotions, leading to inaccurate classifications [42].

VI - Conclusion:

To conclude, being a fundamental aspect of human behaviour, emotions play a crucial role in communication and decision-making. Therefore, accurate emotion detection can offer significant insights into public sentiment and social trends. Our study has sought to compare and evaluate both RNN-GRU and CNN-GRU models for 13 emotion detection in social media tweets. Indeed, our aim was to address existing gaps in predictive methodologies and assess their practical implications in real-world settings. Our findings validate the high efficacy of both our models, with our CNN-GRU model significantly outperforming our RNN-GRU model, achieving higher accuracy across training, testing, and deployment stages. This superior performance is attributed to CNN's capability to capture spatial hierarchies in text data effectively, thereby enhancing its accuracy and robustness. However, both our models faced challenges in handling rare emotion classes, particularly with low-support classes where both showed their weaknesses, leading to the emphasis of the further need of improvements. Therefore, to enhance emotion detection performance, future work should focus on addressing class imbalances through techniques like SMOTE or data augmentation and exploring advanced models and hybrid approaches to further refine accuracy and reliability. However, it is crucial to address ethical considerations when detecting emotions on social media platforms, such as privacy concerns as tweets often contain sensitive personal information, data anonymization and permission obtention to protect users' identities. Finally, the context-dependency of emotional expression complicates accurate detection. Therefore, it is crucial to not train models on imbalanced datasets as they may underrepresent certain emotions or demographic groups, leading to biased results and potential unfair treatment.

VI - References:

[1] University of West Alabama (2019). The Science of Emotion: Exploring the Basics of Emotional Psychology | UWA Online. [online] UWA Online. Available at: https://online.uwa.edu/news/emotional-psychology/.

[2] Dennison, J. (2023). Emotions: functions and significance for attitudes, behaviour, and communication. Migration Studies, 12(1). doi:https://doi.org/10.1093/migration/mnad018.

[3] Purves, D., Augustine, G.J., Fitzpatrick, D., Katz, L.C., Anthony-Samuel LaMantia, McNamara, J.O. and S Mark Williams (2011). Physiological Changes Associated with Emotion. [online] Nih.gov. Available at: https://www.ncbi.nlm.nih.gov/books/NBK10829/.

[4] Cherry, K. (2023). Emotions and types of emotional responses. [online] Verywell Mind. Available at: https://www.verywellmind.com/what-are-emotions-2795178.

[5] Preckel, K., Kanske, P. and Singer, T. (2018). On the interaction of social affect and cognition: empathy, compassion and theory of mind. Current Opinion in Behavioral Sciences, 19, pp.1–6. doi:https://doi.org/10.1016/j.cobeha.2017.07.010.

[6] Basics of Psychology. (2024). Emotions in Decision Making: Impact on Choices. [online] Available at: https://basicsofpsychology.com/emotion-in-decision-making/ [Accessed 9 Aug. 2024].

[7] Lin, Shangyue. (2024). Text emotional analysis in Natural Language Processing. Applied and Computational Engineering. 36. 163-172. 10.54254/2755-2721/36/20230440.

[8] Nandwani, P. and Verma, R. (2021). A review on sentiment analysis and emotion detection from text. Social Network Analysis and Mining, [online] 11(1). doi:https://doi.org/10.1007/s13278-021-00776-6.

[9] Bilal Abu-Salih, Alhabashneh, M., Zhu, D., Albara Awajan, Yazan Alshamaileh, Bashar Al-Shboul and Alshraideh, M. (2023). Emotion detection of social data: APIs comparative study. 9(5), pp.e15926–e15926. doi:https://doi.org/10.1016/j.heliyon.2023.e15926.

[10] Yagnesh P (2024). Customer Insights: The Power of Sentiment Analysis on Social Media Introduction In the digital era, social media platforms have become invaluable sources of information for businesses. Understanding customer perceptions and sentiments expressed on these platforms is crucial for maintaining brand rep. [online] Linkedin.com. Available at: https://www.linkedin.com/pulse/how-does-sentiment-analysis-help-businesses-customer-social-pandya-vlz2f/ [Accessed 9 Aug. 2024].

[11] IBM (2023). What Are Neural Networks? [online] www.ibm.com. Available at: https://www.ibm.com/topics/neural-networks.

[12] IBM. (2024). What Is Machine Learning (ML)? | IBM. [online] Available at: https://www.ibm.com/topics/machine-learning#:~:text=Supervised%20learning%2C%20also%20known%20as.

[13] Purwono, Purwono & Ma'arif, Alfian & Rahmaniar, Wahyu & Imam, Haris & Fathurrahman, Haris Imam Karim & Frisky, Aufaclav & Haq, Qazi Mazhar Ul. (2023). Understanding of Convolutional Neural Network (CNN): A Review. 2. 739-748. 10.31763/ijrcs.v2i4.888.

[14] Kalita, D. (2022). A Brief Overview of Recurrent Neural Networks (RNN). [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/#:~:text=Recurrent%20Neural%20Networks%20(RNNs)%20are.

[15] Nguyen, G., Dlugolinsky, S., Bobák, M., Tran, V., López García, Á., Heredia, I., Malík, P. and Hluchý, L. (2019). Machine learning and deep learning frameworks and libraries for large-scale data mining: A survey. Artificial Intelligence Review, 52(1), pp.77–124. doi:https://doi.org/10.1007/s10462-018-09679-z.

[16] Taye, M.M. (2023). Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions. Computers, [online] 12(5), pp.91–91. doi:https://doi.org/10.3390/computers12050091.

[17] Kristína Machová, Szabóová, M., Ján Paralič and Ján Mičko (2023). Detection of emotion by text analysis using machine learning. Frontiers in Psychology, 14. doi:https://doi.org/10.3389/fpsyg.2023.1190326.

[18] GeeksforGeeks. (2023). Emotion Detection Using Convolutional Neural Networks (CNNs). [online] Available at: https://www.geeksforgeeks.org/emotion-detection-using-convolutional-neural-networks-cnns/.

[19] Cahyani, D.E., Wibawa, A.P., Prasetya, D.D., Gumilar, L., Akhbar, F. and Triyulinar, E.R. (2022). Text-Based Emotion Detection using CNN-BiLSTM. [online] IEEE Xplore. doi:https://doi.org/10.1109/ICORIS56080.2022.10031370.

[20] Bharti, S.K., Varadhaganapathy, S., Gupta, R.K., Shukla, P.K., Bouye, M., Hingaa, S.K. and Mahmoud, A. (2022). Text-Based Emotion Recognition Using Deep Learning Approach. Computational Intelligence and Neuroscience, [online] 2022, p.2645381. doi:https://doi.org/10.1155/2022/2645381.

[21] Shelf (2024). Why Recurrent Neural Networks (RNNs) Dominate Sequential Data Analysis. [online] Shelf. Available at: https://shelf.io/blog/recurrent-neural-networks/.

[22] Yohanes, D., Putra, J.S., Filbert, K., Suryaningrum, K.M. and Saputri, H.A. (2023). Emotion Detection in Textual Data using Deep Learning. Procedia Computer Science, [online] 227, pp.464–473. doi:https://doi.org/10.1016/j.procs.2023.10.547.

[23] Kaggle.com (n.d.). emotion analysis based on text. [online] Available at: https://www.kaggle.com/datasets/simaanjali/emotion-analysis-based-on-text.

[24] Ghosh, K., Bellinger, C., Corizzo, R., Branco, P., Krawczyk, B. and Japkowicz, N. (2022). The class imbalance problem in deep learning. Machine Learning. doi:https://doi.org/10.1007/s10994-022-06268-8.

[25] Linkedin.com. (2024). How can you ensure your ML model is robust when working with samples? [online] Available at: https://www.linkedin.com/advice/1/how-can-you-ensure-your-ml-model-robust-when-working-jzvrc [Accessed 9 Aug. 2024].

[26] Johnson, J.M. and Khoshgoftaar, T.M. (2019). Survey on deep learning with class imbalance. Journal of Big Data, 6(1). doi:https://doi.org/10.1186/s40537-019-0192-5.

[27] Yin, Z., Zhao, M., Wang, Y., Yang, J. and Zhang, J. (2017). Recognition of emotions using multimodal physiological signals and an ensemble deep learning model. Computer Methods and Programs in Biomedicine, 140, pp.93–110. doi:https://doi.org/10.1016/j.cmpb.2016.12.005.

[28] Van Giffen, B., Herhausen, D. and Fahse, T. (2022). Overcoming the pitfalls and perils of algorithms: A classification of machine learning biases and mitigation methods. Journal of Business Research, 144(1), pp.93–106. doi:https://doi.org/10.1016/j.jbusres.2022.01.076.

[29] Khan, A.A., Chaudhari, O. and Chandra, R. (2023). A Review of Ensemble Learning and Data Augmentation Models for Class Imbalanced problems: Combination, Implementation and Evaluation. Expert Systems with Applications, [online] 244, p.122778. doi:https://doi.org/10.1016/j.eswa.2023.122778.

[30] Mobarak Abumohsen, Abumihsan, A., Amani Yousef Owda and Majdi Owda (2024). Hybrid Machine Learning Model Combining of CNN-LSTM-RF for Time Series Forecasting of SPG. e-Prime - Advances in Electrical Engineering Electronics and Energy, pp.100636–100636. doi:https://doi.org/10.1016/j.prime.2024.100636.

[31] Zhao, Z., Alzubaidi, L., Zhang, J., Duan, Y. and Gu, Y. (2024). A comparison review of transfer learning and self-supervised learning: Definitions, applications, advantages and limitations. Expert Systems with Applications, [online] 242, p.122807. doi:https://doi.org/10.1016/j.eswa.2023.122807.

[32] Gorodetski, M. (2021). Hyperparameter Tuning Methods — Grid, Random or Bayesian Search? [online] Medium. Available at: https://towardsdatascience.com/bayesian-optimization-for-hyperparameter-tuning-how-and-why-655b0ee0b399.

[33] Hernández, A. and Amigó, J.M. (2021). Attention Mechanisms and Their Applications to Complex Systems. Entropy, [online] 23(3), p.283. doi:https://doi.org/10.3390/e23030283.

[34] Doris, Lucas & Potter, Kaledio. (2024). Continuous Monitoring and Improvement: Implement continuous monitoring of AI models to detect and correct issues in real-time. i-manager s Journal on Artificial Intelligence & Machine Learning.

[35] ForumCosmos (2023). Ethical Considerations in Data Privacy and Security. [online] Medium. Available at: https://medium.com/@armaanakhan91/ethical-considerations-in-data-privacy-and-security-1874a10061f0#:~:text=Using%20data%20beyond%20its%20original [Accessed 9 Aug. 2024].

[36] IMPERVA (2019). What is Data Anonymization | Pros, Cons & Common Techniques | Imperva. [online] Learning Center. Available at: https://www.imperva.com/learn/data-security/anonymization/.

[37] Zimmer, Michael & Proferes, Nicholas. (2014). A Topology of Twitter Research: Disciplines, Methods, and Ethics. Aslib Journal of Information Management. 66. 10.1108/AJIM-09-2013-0083.

[38] Marcello Ienca (2023). On Artificial Intelligence and Manipulation. On Artificial Intelligence and Manipulation. doi:https://doi.org/10.1007/s11245-023-09940-3.

[39] Confessore, N. (2018). Cambridge Analytica and Facebook: The Scandal and the Fallout So Far. The New York Times. [online] 4 Apr. Available at: https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html.

[40] Siddique, S., Haque, M.A., George, R., Gupta, K.D., Gupta, D. and Faruk, M.J.H. (2024). Survey on Machine Learning Biases and Mitigation Techniques. Digital, [online] 4(1), pp.1–68. doi:https://doi.org/10.3390/digital4010001.

[41] Khare, S.K., Blanes-Vidal, V., Nadimi, E.S. and Acharya, U.R. (2024). Emotion recognition and artificial intelligence: A systematic review (2014–2023) and research recommendations. Information Fusion, [online] 102, p.102019. doi:https://doi.org/10.1016/j.inffus.2023.102019.

[42] Guo, R., Guo, H., Wang, L., Chen, M., Yang, D. and Li, B. (2024). Development and application of emotion recognition technology — a systematic literature review. BMC Psychology, 12(1). doi:https://doi.org/10.1186/s40359-024-01581-4.

[43] Mohammad, Saif & Turney, Peter. (2013). Crowdsourcing a Word-Emotion Association Lexicon. Computational Intelligence. 29. 10.1111/j.1467-8640.2012.00460.x.