Fine-Tuning MentalBERT for Sentiment Classification in Mental Health Texts

Dominic Boy Almazan
ITE-IV, BSIT
Jose Rizal University
Mandaluyong City, Philippines
dominicboy.almazan@my.jru.edu

***Abstract*— This study focuses on fine-tuning the MentalBERT model to classify mental health–related tweets as either Normal or Suicidal. The dataset, sourced from Kaggle, was preprocessed through cleaning, tokenization, and encoding to prepare it for training. The model was accessed through the Hugging Face API, which provides free access to pretrained transformer models for research and development. Fine-tuning was performed using the Hugging Face Trainer API with the AdamW optimizer and cross-entropy loss function. Ten experiments were conducted by adjusting key hyperparameters, including learning rate, weight decay, and number of epochs, to analyze their impact on model performance. Results showed that the best configuration, using a learning rate of 5e-5, weight decay of 0.1, and 5 training epochs, achieved an F1-score of 0.983, outperforming the baseline model with an F1-score of 0.977. The findings demonstrate that effective preprocessing, accessible pretrained models, and systematic hyperparameter tuning can significantly enhance the performance of domain-specific transformer models like MentalBERT in detecting suicidal ideation from textual data.**

Keywords*— MentalBERT, Sentiment Analysis, Fine-Tuning, Natural Language Processing, Hyperparameter Optimization, Mental Health

I. INTRODUCTION

Mental health analysis using Natural Language Processing (NLP) has become increasingly important in today's digital age, where many people express their emotions and experiences through online platforms. Social media posts, forums, and online journals often contain valuable clues about a person's mental and emotional state. Detecting early signs of distress or suicidal thoughts from written language can help provide timely support and intervention, potentially saving lives. Recent advances in NLP have made it possible to analyze such text automatically and accurately. Among these advances, transformer-based language models such as BERT and its variations have brought major improvements in understanding the meaning and context of human language [1]. These models can interpret not only the words themselves but also the deeper emotions and intentions behind them, which is essential when studying mental health communication.

MentalBERT is one of these advanced models and was specifically developed for applications related to mental health. It is a domain-specific version of BERT that has been trained on large collections of text dealing with psychological well-being, emotional support, and mental health discussions [2]. Because it has been exposed to this specialized data, MentalBERT is better at recognizing subtle expressions of sadness, anxiety, or hopelessness that might not be detected by general-purpose models. This capability makes it suitable for analyzing posts or statements where people may express emotional pain indirectly. By fine-tuning the model on a smaller, task-specific dataset, such as a collection of sentences labeled as Normal or Suicidal, MentalBERT can adapt to identify patterns that are unique to a particular population or study environment [3].

This paper presents the process of fine-tuning MentalBERT for binary sentiment classification, focusing on the differentiation between normal and suicidal statements. The experiment aimed to examine how various model configurations, particularly the learning rate, weight decay, and number of training epochs, influence performance. These hyperparameters were carefully adjusted to improve model accuracy and reliability. The study also highlights the importance of fine-tuning as a way to optimize existing models for specialized tasks, demonstrating how small adjustments in training conditions can lead to significant improvements in the detection of mental health indicators in written text.

II. METHODOLOGY

A. Dataset Description

The dataset used in this study was obtained from Kaggle, titled "Sentiment Analysis for Mental Health" [4]. It consists of user-generated tweets that express a variety of emotions and mental states, making it a suitable resource for research on mental health–related sentiment analysis. The dataset was selected because it reflects real-world language used in social media, where individuals often express their thoughts and emotional well-being openly and informally. This kind of data provides a valuable foundation for training models to recognize the linguistic patterns associated with normal and potentially suicidal expressions.

The dataset is structured with three main columns: ID, Statement, and Tweet. Each serves a specific role in organizing and preparing the data for the sentiment classification task.

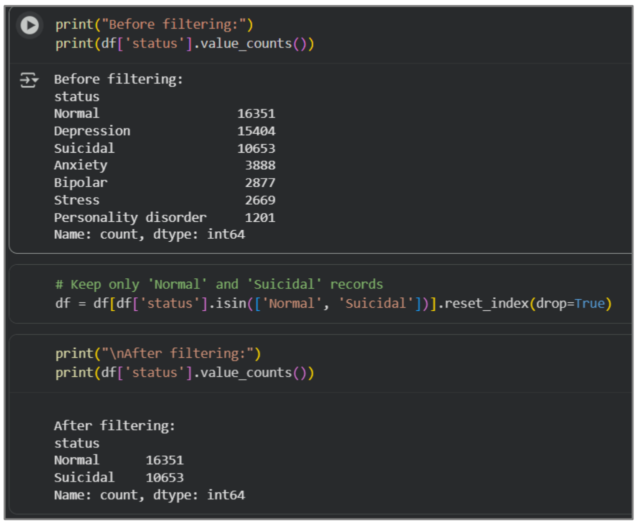
- ID
  This column serves as the unique identifier for each record in the dataset. It ensures proper indexing and tracking of data instances during preprocessing, model training, and evaluation. Each ID corresponds to one unique tweet entry.
- Status
  The Status column contains the sentiment label assigned to each tweet. The original dataset includes five categories that represent different emotional or psychological conditions expressed in the text. These categories are designed to capture a wide range of mental health–related sentiments. However, for the purpose of this study, only two categories were utilized Normal and Suicidal. This binary classification setup simplifies the model's objective to distinguish between typical emotional expressions and those indicating potential suicidal intent. Tweets labeled under other categories were excluded to maintain a clear focus on detecting critical mental health cues.
- Statement
  The Statement column holds the actual text data collected from Twitter. Each entry reflects real social media content where individuals express their daily thoughts, emotions, and mental states. The text often includes informal language, emojis, abbreviations, and grammatical variations, making it representative of authentic online communication. This column serves as the main input for the MentalBERT model after preprocessing and tokenization.

By focusing on the Normal and Suicidal classes, the dataset provides a meaningful foundation for studying how fine-tuned transformer models can differentiate between neutral and high-risk mental health expressions within real-world social media data..

B. Data Preprocessing

Before training the model, several preprocessing steps were performed to prepare the dataset for fine-tuning. These steps ensured that the input data was properly formatted and consistent with the requirements of transformer-based models such as MentalBERT.

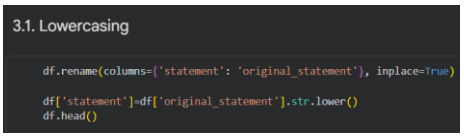
The first step involved filtering the dataset to retain only the relevant classes Normal and Suicidal from the original five sentiment labels. This process simplified the problem into a binary classification task. Once the filtered data was obtained, the Status column was encoded into numerical form using label encoding, where Normal was represented as 0 and Suicidal as 1. This numerical conversion was necessary because machine learning models operate on numeric values rather than textual labels.



```
print("Before filtering:")
print(df['status'].value_counts())

Before filtering:
status
Normal                  16351
Depression              15404
Suicidal                10653
Anxiety                  3888
Bipolar                  2877
Stress                   2669
Personality disorder     1201
Name: count, dtype: int64

# Keep only 'Normal' and 'Suicidal' records
df = df[df['status'].isin(['Normal', 'Suicidal'])].reset_index(drop=True)

print("\nAfter filtering:")
print(df['status'].value_counts())

After filtering:
status
Normal      16351
Suicidal    10653
Name: count, dtype: int64
```

Figure 1. Snippet Code of keeping the 'Normal' and 'Suicidal' in the column of 'status'

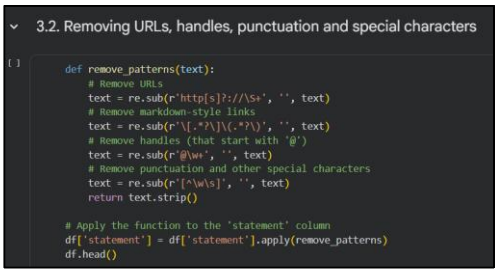
After labeling, text cleaning was applied to the Tweet column to remove unnecessary elements such as special characters, punctuation marks, repeated whitespace, and URLs. This step helped standardize the language and minimize noise in the input text. The cleaned tweets were then converted to lowercase to ensure that the model treated words like "Sad" and "sad" as the same token.



3.1. Lowercasing

```
df.rename(columns={'statement': 'original_statement'}, inplace=True)

df['statement']=df['original_statement'].str.lower()
df.head()
```

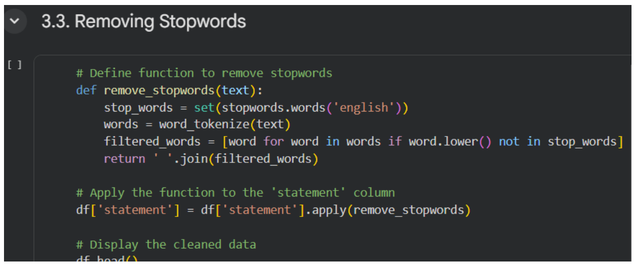
Figure 2. Snippet Code of Lowercasing



3.2. Removing URLs, handles, punctuation and special characters

```
def remove_patterns(text):
    # Remove URLs
    text = re.sub(r'http[s]?://\S+', '', text)
    # Remove markdown-style links
    text = re.sub(r'\[.*?\]\(.*?\)', '', text)
    # Remove handles (that start with '@')
    text = re.sub(r'@\w+', '', text)
    # Remove punctuation and other special characters
    text = re.sub(r'[^\w\s]', '', text)
    return text.strip()

# Apply the function to the 'statement' column
df['statement'] = df['statement'].apply(remove_patterns)
df.head()
```

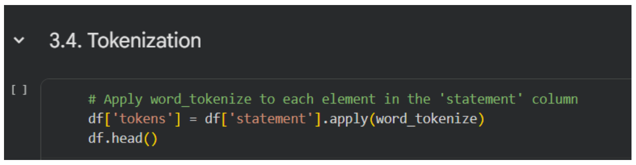
Figure 3. Snippet Code of Removing URL,punctuations and special characters.



3.3. Removing Stopwords

```
# Define function to remove stopwords
def remove_stopwords(text):
    stop_words = set(stopwords.words('english'))
    words = word_tokenize(text)
    filtered_words = [word for word in words if word.lower() not in stop_words]
    return ' '.join(filtered_words)

# Apply the function to the 'statement' column
df['statement'] = df['statement'].apply(remove_stopwords)

# Display the cleaned data
df.head()
```

Figure 4. Snippet Code of Removing Stopwords

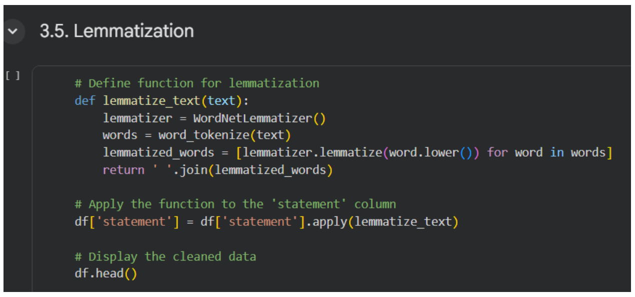


3.4. Tokenization

```
# Apply word_tokenize to each element in the 'statement' column
df['tokens'] = df['statement'].apply(word_tokenize)
df.head()
```

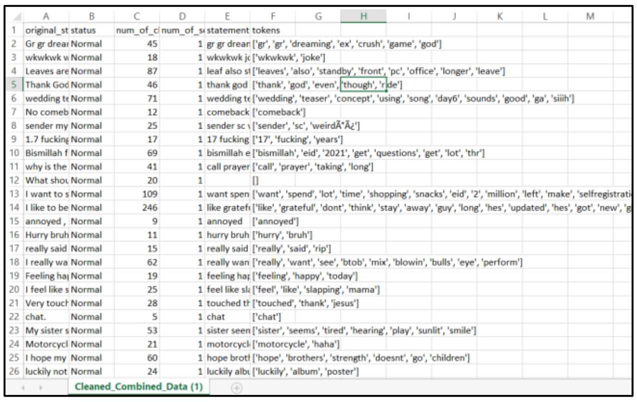
Figure 5. Snippet Code of Removing Tokenization

Next, the text data was tokenized using the MentalBERT tokenizer, which splits each tweet into individual subword tokens and converts them into corresponding integer IDs. The tokenizer also added special tokens ([CLS] and [SEP]) required by the BERT architecture and truncated or padded each sequence to a fixed maximum length of 128 tokens. This step ensured uniform input size across all samples, improving training stability.


3.5. Lemmatization

```
# Define function for lemmatization
def lemmatize_text(text):
    lemmatizer = WordNetLemmatizer()
    words = word_tokenize(text)
    lemmatized_words = [lemmatizer.lemmatize(word.lower()) for word in words]
    return ' '.join(lemmatized_words)

# Apply the function to the 'statement' column
df['statement'] = df['statement'].apply(lemmatize_text)

# Display the cleaned data
df.head()
```

Figure 5. Snippet Code of Removing Tokenization



	original_st	status	num_of_c	num_of_s	statement	tokens
2	Gr gr drear	Normal	45	1	gr gr drean	['gr', 'gr', 'dreaming', 'ex', 'crush', 'game', 'god']
3	wkwkwk w	Normal	18	1	wkwkwk jc	['wkwkwk', 'joke']
4	Leaves are	Normal	87	1	leaf also st	['leaves', 'also', 'standby', 'front', 'pc', 'office', 'longer', 'leave']
5	Thank God	Normal	46	1	thank god	['thank', 'god', 'even', 'though', 'ride']
6	wedding te	Normal	71	1	wedding te	['wedding', 'teaser', 'concept', 'using', 'song', 'day6', 'sounds', 'good', 'ga', 'siiih']
7	No comeb	Normal	12	1	comeback	['comeback']
8	sender my	Normal	25	1	sender sc v	['sender', 'sc', 'weirdÃ°Ã¿']
9	1.7 fucking	Normal	17	1	17 fucking	['17', 'fucking', 'years']
10	Bismillah f	Normal	69	1	bismillah e	['bismillah', 'eid', '2021', 'get', 'questions', 'get', 'lot', 'thr']
11	why is the	Normal	41	1	call prayer	['call', 'prayer', 'taking', 'long']
12	What shou	Normal	20	1		[]
13	I want to s	Normal	109	1	want spen	['want', 'spend', 'lot', 'time', 'shopping', 'snacks', 'eid', '2', 'million', 'left', 'make', 'selfregistrati
14	I like to be	Normal	246	1	like gratefı	['like', 'grateful', 'dont', 'think', 'stay', 'away', 'guy', 'long', 'hes', 'updated', 'hes', 'got', 'new', 'g
15	annoyed ,	Normal	9	1	annoyed	['annoyed']
16	Hurry bruh	Normal	11	1	hurry bruh	['hurry', 'bruh']
17	really said	Normal	15	1	really said	['really', 'said', 'rip']
18	I really wa	Normal	62	1	really wan	['really', 'want', 'see', 'btob', 'mix', 'blowin', 'bulls', 'eye', 'perform']
19	Feeling ha	Normal	19	1	feeling hap	['feeling', 'happy', 'today']
20	I feel like s	Normal	25	1	feel like sla	['feel', 'like', 'slapping', 'mama']
21	Very touch	Normal	28	1	touched th	['touched', 'thank', 'jesus']
22	chat.	Normal	5	1	chat	['chat']
23	My sister s	Normal	53	1	sister seen	['sister', 'seems', 'tired', 'hearing', 'play', 'sunlit', 'smile']
24	Motorcycl	Normal	21	1	motorcycl	['motorcycle', 'haha']
25	I hope my	Normal	60	1	hope brotl	['hope', 'brothers', 'strength', 'doesnt', 'go', 'children']
26	luckily not	Normal	24	1	luckily albı	['luckily', 'album', 'poster']

Figure 6. Overview of the Dataset after applying the Pre Processing

After the data cleaning and preprocessing stages, the dataset was refined to include six key columns: original_statement, status, number_of_characters, number_of_sentences, statement, and tokens. The original_statement retains the unprocessed tweet text, serving as a reference for the initial raw data. The status column contains the assigned sentiment label (Normal or Suicidal), which guides the model's classification task. The number_of_characters and number_of_sentences columns provide basic linguistic features that help describe the length and structure of each text sample. The statement column holds the cleaned and standardized version of the tweet used as model input, while the tokens column contains the tokenized representation of the text after applying the MentalBERT tokenizer. Collectively, these columns ensure that both the raw and processed forms of data are preserved, supporting effective training, validation, and interpretability of the fine-tuning process.

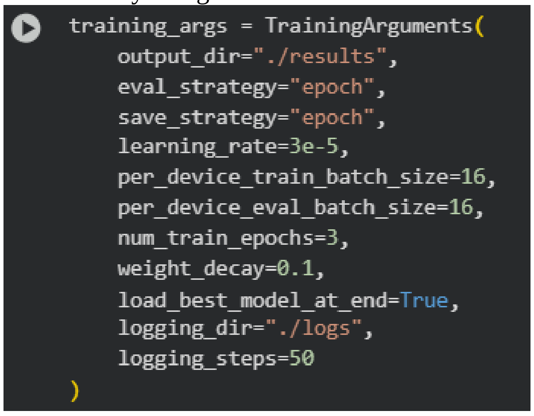
C. Model Setup and Fine-Tuning Process

After the dataset was cleaned, encoded, and tokenized, it was divided into training and testing subsets to support effective model evaluation. Approximately 80% of the data was used for training, while the remaining 20% was reserved for testing. Each subset was then converted into a format compatible with the Hugging Face Trainer API, which simplified the overall workflow for training, validation, and performance monitoring. To further enhance efficiency, a DataLoader was used to batch and shuffle the data automatically, optimizing GPU memory utilization and ensuring that each training epoch presented the model with varied input sequences. These steps collectively prepared the dataset for fine-tuning while preserving the contextual and linguistic integrity of each tweet.

The experiment utilized the MentalBERT model, a domain-specific variant of BERT pretrained on mental health–related text. This model was selected because of its ability to understand emotional and psychological expressions common in mental health discourse. The fine-tuning process aimed to adapt MentalBERT's pretrained parameters to the binary classification task identifying whether a tweet was Normal or Suicidal.

Several hyperparameters were adjusted across different experiments to analyze their influence on model performance. These included:

- Learning Rate - varied between 2e-5 and 5e-5 to evaluate the trade-off between convergence speed and model stability.
- Weight Decay - adjusted between 0.01 and 0.1 to prevent overfitting by controlling the magnitude of weight updates.
- Number of Epochs - tested between 2 and 5 epochs to observe how extended training affected accuracy and generalization



```
training_args = TrainingArguments(
    output_dir="./results",
    eval_strategy="epoch",
    save_strategy="epoch",
    learning_rate=3e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=3,
    weight_decay=0.1,
    load_best_model_at_end=True,
    logging_dir="./logs",
    logging_steps=50
)
```

Figure 7. Snippet Code of Hyperparmeter

During training, evaluation was conducted at the end of each epoch using the F1-score as the main metric for selecting the best-performing model. Other metrics such as accuracy, precision, and recall were also computed to assess classification balance and consistency.

All experiment runs were logged and tracked through TensorBoard and Weights & Biases (W&B) for performance visualization and comparison. This monitoring helped identify trends such as overfitting, optimal learning rate regions, and the most effective regularization parameters.

The trained models were then saved locally for further evaluation and testing on unseen text samples. The combination of systematic hyperparameter tuning and domain-specific fine-tuning enabled the MentalBERT model to effectively recognize subtle emotional differences between Normal and Suicidal statements, reflecting strong potential for practical applications in automated mental health assessment systems.

III. RESULTS

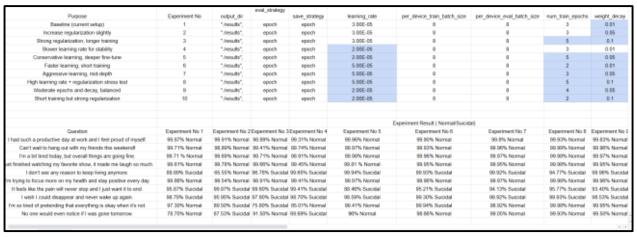
The fine-tuned MentalBERT model was evaluated to determine how different hyperparameter configurations affected its ability to classify tweets as Normal or Suicidal. The evaluation metrics included evaluation loss, accuracy, precision, recall, and F1-score, which together provide a balanced view of the model's predictive performance.

The baseline configuration already achieved strong results, with an F1-score of 0.977, confirming that MentalBERT's pretrained knowledge provides high performance even with minimal tuning. Increasing weight decay from 0.01 to 0.05 (Experiment 2) produced slightly more consistent outcomes, indicating better regularization and improved control over overfitting.

Table 1. Summary of Hyperparameter Experiment and Result

Category	Experiment	Accuracy	Precision	Recall	F1-Score
Baseline	1	0.978	0.977	0.977	0.976
Best Result	8	0.983	0.9829	0.983	0.983
Lowest Result (Still Decent)	4	0.9483	0.948	0.9483	0.948

When both weight decay and training epochs were increased (Experiment 3), performance improved further, reaching an F1-score of 0.9804. This suggests that extended training allows the model to refine its understanding of contextual and emotional cues, while stronger regularization helps maintain generalization.



Purpose	Experiment No	output_dir	eval_strategy	save_strategy	learning_rate	per_device_train_batch_size	per_device_eval_batch_size	num_train_epochs	weight_decay
Baseline (current setup)	1	"./results",	epoch	epoch	3.00E-05	8	8	3	0.01
Increase regularization slightly	2	"./results",	epoch	epoch	3.00E-05	8	8	3	0.05
Strong regularization, longer training	3	"./results",	epoch	epoch	3.00E-05	8	8	5	0.1
Slower learning rate for stability	4	"./results",	epoch	epoch	2.00E-05	8	8	3	0.01
Conservative learning, deeper fine-tune	5	"./results",	epoch	epoch	2.00E-05	8	8	5	0.05
Faster learning, short training	6	"./results",	epoch	epoch	5.00E-05	8	8	2	0.01
Aggressive learning, mid-depth	7	"./results",	epoch	epoch	5.00E-05	8	8	3	0.05
High learning rate + regularization stress test	8	"./results",	epoch	epoch	5.00E-05	8	8	5	0.1
Moderate epochs and decay, balanced	9	"./results",	epoch	epoch	2.00E-05	8	8	4	0.05
Short training but strong regularization	10	"./results",	epoch	epoch	2.00E-05	8	8	2	0.1

Experiment Result (Normal/Suicidal)

Question	Experiment No 1	Experiment No 2	Experiment No 3	Experiment No 4	Experiment No 5	Experiment No 6	Experiment No 7	Experiment No 8	Experiment No 9
I had such a productive day at work and I feel proud of myself.	99.87% Normal	99.91% Normal	98.89% Normal	99.31% Normal	99.96% Normal	99.90% Normal	99.8% Normal	99.93% Normal	99.83% Normal
Can't wait to hang out with my friends this weekend!	99.71% Normal	98.89% Normal	99.41% Normal	99.74% Normal	99.97% Normal	99.93% Normal	99.96% Normal	99.99% Normal	99.96% Normal
I'm a bit tired today, but overall things are going fine.	99.71 % Normal	99.69% Normal	99.71% Normal	98.91% Normal	99.98% Normal	99.96% Normal	99.97% Normal	99.98% Normal	99.97% Normal
ust finished watching my favorite show, it made me laugh so much.	99.81% Normal	99.78% Normal	99.66% Normal	99.45% Normal	99.81 % Normal	99.95% Normal	99.95% Normal	99.98% Normal	99.95% Normal
I don't see any reason to keep living anymore.	88.89% Suicidal	65.55% Normal	98.78% Suicidal	99.65% Suicidal	99.94% Suicidal	99.93% Suicidal	99.92% Suicidal	94.77% Suicidal	99.96% Suicidal
I'm trying to focus more on my health and stay positive every day.	99.86% Normal	99.54% Normal	98.91% Normal	99.41% Normal	99.97% Normal	99.96% Normal	99.97% Normal	99.98% Normal	99.96% Normal
It feels like the pain will never stop and I just want it to end.	95.87% Suicidal	99.87% Suicidal	99.60% Suicidal	99.41% Suicidal	88.46% Suicidal	95.21% Suicidal	94.13% Suicidal	95.77% Suicidal	93.40% Suicidal
I wish I could disappear and never wake up again.	98.78% Suicidal	95.95% Suicidal	97.80% Suicidal	98.70% Suicidal	98.59% Suicidal	99.30% Suicidal	98.92% Suicidal	99.93% Suicidal	98.53% Suicidal
I'm so tired of pretending that everything is okay when it's not.	97.30% Normal	89.50% Suicidal	75.80% Suicidal	85.01% Normal	99.41% Normal	99.94% Suicidal	98.92% Normal	99.98% Normal	99.95% Normal
No one would even notice if I was gone tomorrow.	78.70% Normal	87.53% Suicidal	91.50% Normal	69.69% Suicidal	96% Normal	98.66% Normal	99.05% Normal	99.93% Normal	99.50% Normal

Figure 8. Experimentation and Result Logs

Experiment 8 also demonstrated stable performance across evaluation metrics, confirming that the model achieved an optimal balance between speed of learning and regularization. In contrast, configurations with fewer epochs (Experiments 6 and 10) showed minor decreases in accuracy and recall, suggesting that the model benefited from more complete exposure to the dataset.

Overall, the results indicate that fine-tuning MentalBERT with a higher learning rate (5e-5), stronger regularization (weight decay=0.1), and longer training duration (5 epochs) provides the best trade-off between accuracy, generalization, and training efficiency. The model successfully identified tweets expressing suicidal ideation with high precision and recall, demonstrating the capability of domain-specific transformer models in supporting automated mental health text analysis.

IV. DISCUSSION

The workflow of this study began with a detailed preprocessing phase to prepare the dataset for fine-tuning. Only two sentiment classes Normal and Suicidal were retained from the original five labels to simplify the task into binary classification. Each tweet was cleaned by removing unnecessary symbols, punctuation, and excess whitespace, and all text was converted to lowercase for uniformity. The MentalBERT tokenizer was then applied to break sentences into subword tokens, add special tokens, and standardize input length through padding and truncation. The dataset was divided into 80% for training and 20% for testing, formatted for the Hugging Face Trainer API, and processed with a DataLoader to efficiently batch and shuffle the data during each epoch.

Access to the MentalBERT model required authentication through the Hugging Face API, which is freely available for researchers and developers. This API allows direct access to pretrained models, enabling users to fine-tune them easily within their local environment. Using this platform simplified the setup process and made model experimentation more manageable. Once the model was loaded, training was performed using the AdamW optimizer and cross-entropy loss function, both commonly used for transformer-based architectures. Evaluation occurred at the end of each epoch, measuring accuracy, precision, recall, and F1-score to monitor model progress and detect signs of overfitting or underfitting.

The experimentation revealed that small adjustments to hyperparameters significantly influenced the model's performance. The baseline model achieved an F1-score of 0.977, while the best configuration with a higher learning rate (5e-5), stronger weight decay (0.1), and longer training (5 epochs) achieved an F1-score of 0.983, showing the value of fine-tuning. In contrast, using a lower learning rate of 2e-5 led to slower convergence and a reduced F1-score of 0.9483. These results confirm that combining proper preprocessing, free API access to pretrained models, and systematic hyperparameter optimization can greatly improve MentalBERT's ability to recognize suicidal expressions in textual data.

V. CONCLUSION

This study successfully demonstrated the fine-tuning of the MentalBERT model for binary sentiment classification of mental health–related text. Through systematic data preprocessing, structured experimentation, and hyperparameter tuning, the model achieved a high level of accuracy and reliability in distinguishing between *Normal* and *Suicidal* tweets. The experiment validated that proper data cleaning, tokenization, and parameter optimization are essential steps in improving model performance for sensitive NLP applications such as mental health monitoring.

The baseline experiment already produced strong performance with an F1-score of 0.977, indicating that MentalBERT's pretrained knowledge is highly effective for understanding emotional content. Further adjustments to the learning rate, weight decay, and number of epochs resulted in noticeable improvements, with the best-performing configuration reaching an F1-score of 0.983. The results highlight that increasing the number of training epochs and applying stronger regularization enable the model to learn deeper contextual representations without overfitting.

For future work, several areas can be explored to enhance both performance and generalization. First, increasing the number of epochs beyond the current limit may further improve accuracy, allowing the model to better capture nuanced emotional patterns. Second, experimenting with different pretrained transformer models, such as RoBERTa, DistilBERT, or ALBERT, could provide valuable insights into how various architectures perform on the same dataset. Comparing these results would help identify the most efficient and context-sensitive model for mental health–related sentiment classification.

Overall, the findings of this study emphasize that with proper preprocessing, accessible pretrained models via the Hugging Face API, and strategic fine-tuning, transformer-based architectures like MentalBERT can be powerful tools in supporting automated systems for early detection of mental health signals in online text.

VI. REFERENCES

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proc. NAACL-HLT*, 2019 Available : https://aclanthology.org/N19-1423.pdf
.

[2] M. Nguyen et al., "MentalBERT: Publicly Available Pretrained Language Models for Mental Healthcare," *arXiv preprint arXiv:2205.03767*, 2022.
Available: https://arxiv.org/pdf/2110.15621

[3] Y. Liu, X. He, and F. Wang, "Fine-tuning Pretrained Language Models for Mental Health Prediction," *IEEE Access*, vol. 10, pp. 95122–95133, 2022. Available:
https://pmc.ncbi.nlm.nih.gov/articles/PMC12072099/

[4] S. Sarkar, *"Sentiment Analysis for Mental Health,"* Kaggle, 2022. Available:
https://www.kaggle.com/datasets/suchintikasarkar/sentiment-analysis-for-mental-health