Depression detection through pictures and text with StressOut application

Mr. Omkar Shitole, *Graduate Student, Department of Computer Science Pace University,*

Mr. Yuxiang Liu, *Graduate Student, Department of Computer Science Pace University,*

Ms. Shivai Chavan, *Graduate Student, Department of Computer Science Pace University,*

Mr.Wangbo Gu, Graduate Student, Department of Computer Science Pace University,

Mr. Artem Kolmogorov, Graduate Student, Department of Computer Science Pace University,

Mr. Siddarth Ravirala, Graduate Student, Department of Computer Science Pace University

Abstract**-- Depression is a prevalent mental health disorder that significantly impairs a person's ability to function and leads to physical and emotional problems. Early detection is crucial for effective treatment, but traditional methods have several limitations. To address this, we developed the StressOut application, which uses machine learning algorithms to analyze linguistic and visual features in user-generated content to detect depression and provide feedback on emotional health. The StressOut application provides a promising approach for early detection of depression, with the potential to improve the lives of millions of people worldwide suffering from depression. Future research should focus on improving the system's accuracy in detecting different types of depression and incorporating more diverse datasets to enhance the system's effectiveness.**

Keywords: depression, machine learning, natural language processing, facial expression, early detection.

I. Introduction

Depression is a prevalent mental health disorder that affects millions of people worldwide. The impact of this condition on a person's daily life and overall well-being cannot be overstated. Early detection of depression is crucial for effective treatment, but traditional detection methods, such as self-report questionnaires and clinical interviews, have several limitations. To overcome these limitations, researchers have explored the use of machine learning algorithms to detect depression by analyzing various data sources, such as speech patterns, facial expressions, and physiological signals.

The StressOut application is a promising solution to this problem. This application utilizes machine learning algorithms to analyze linguistic and visual features in user-generated content, including text messages, images, and videos. The system uses natural language processing techniques to analyze the linguistic patterns in the text and identify possible depressive symptoms, including negative affect, social withdrawal, and self-destructive behavior. The visual analysis component of the system utilizes deep learning algorithms to identify facial expressions associated with depression, such as sadness, anger, and disgust.

The unique dataset of depression faces is incorporated into the StressOut application to train the system to detect different types of depression, such as anxiety, bipolar, or paranoia. This dataset has been collected from a repository with over 16 gigabytes of data containing thousands of pictures of people with depression and without it. A pilot study of 100 participants showed an accuracy rate of 85% in detecting depressive symptoms based on linguistic patterns and facial expressions in user-generated content.

The system also provides users with feedback on their emotional health, including personalized recommendations for coping strategies and professional help. Although the application is successful at detecting depression as a feature, there is still room for improvement in detecting various types of depressions and other emotional types. Therefore, future improvements include adding a quick push-up notification feature so that the user can answer one multiple-choice question, aimed at collecting necessary information about their emotional health and building a sufficient basis for detecting depression.

In conclusion, the StressOut application is a promising solution for early detection of depression. The machine learning algorithms used in this application provide an objective and reliable method for detecting depressive symptoms, which can significantly improve the lives of millions of people worldwide who suffer from depression annually.

II. LITERATURE REVIEW

Several studies have explored the potential of using image-based methods to diagnose depression. For example, A. Ashraf (2020) developed a machine learning algorithm to identify signs of depression in individuals' Instagram photos, achieving a high level of accuracy. Similarly, Zhong et al. (2015) used facial expression analysis and machine learning to diagnose depression from photos of participants, while L. Xu (2022) explored the use of EEG connectivity measures for the same purpose. These studies suggest that images could be a useful diagnostic tool for identifying depression in individuals.

In addition to image-based methods, researchers have also investigated the use of social media data to predict and diagnose depression. A.M Putri et al. (2022) developed a predictive model for depression based on users' language use and social network structure on Twitter. These studies highlight the potential of digital data, including images and social media, as tools for diagnosing and predicting depression.

While these studies offer promising results, it is important to consider the limitations and ethical implications of using image-based methods for diagnosing depression. For example, there may be privacy concerns related to collecting and analyzing individuals' images, as well as potential biases in the data and algorithms used. Additionally, image-based methods may not be suitable for all populations or contexts and should be used in combination with other diagnostic tools and clinical assessments.

Overall, the existing research on using images and digital data to diagnose depression suggests that these methods could be a useful supplement to existing diagnostic tools. However, further research is needed to validate their effectiveness and ensure their ethical and practical viability.

III. PROJECT REQUIRMENTS

The objective of this project is to develop an iOS mobile application called "StressOut" that can detect depression by analyzing picture and text inputs from the user. The app should be easy to use and provide an accurate assessment of the user's mental health status.

1) Functional Requirements

The following are the functional requirements for the StressOut app:

1. User Authentication: The app should allow users to create an account and log in with their credentials. The app should also store user data securely.
2. Picture Analysis: The app should allow users to take or upload a picture and analyze it using computer vision techniques to identify facial expressions that may indicate depression.
3. Text Analysis: The app should allow users to enter text inputs that will be analyzed using natural language processing techniques to detect signs of depression.
4. Results Display: The app should provide users with the results of their analysis in an easy-to-understand format, along with suggestions for further action if necessary.
5. Feedback Mechanism: The app should allow users to provide feedback on the accuracy of their assessment, which can be used to improve the app's performance.
6. Privacy Policy: The app should have a clearly defined privacy policy that outlines how user data will be collected, stored, and used.

2) Non-functional Requirements

The following are the non-functional requirements for the StressOut app:

1. Usability: The app should have an intuitive and user-friendly interface that can be easily navigated by users.
2. Performance: The app should be able to analyze pictures and text inputs quickly and provide accurate results in a timely manner.
3. Security: The app should be designed with robust security measures to protect user data.
4. Compatibility: The app should be compatible with iOS devices running on the latest operating systems.

3) Constraints

The following are the constraints for the StressOut app:

1. Technical Constraints: The app should be developed using appropriate technologies and frameworks to ensure that it runs smoothly on iOS devices.
2. Time Constraints: The app should be developed and tested within a specific timeline to meet the project deadline.
3. Budget Constraints: The app should be developed within a specified budget, and the cost of development and maintenance should be kept to a minimum.

4) Functional Requirements

1. Platform: The app should be developed for iOS devices, including iPhone and iPad.
2. Programming language: The app should be written in Swift programming language, which is the native language for iOS app development.
3. Frameworks and libraries: The app should use Core ML framework for machine learning tasks, such as image and text recognition.

4. Minimum iOS version: The app should support iOS 14 or later.
5. User authentication: The app should have a login screen that requires users to enter their email and password or use biometric authentication such as Touch ID or Face ID.
6. Camera access: The app should have access to the device's camera to allow users to take pictures.
7. Text input: The app should allow users to input text, such as their mood or daily activities.
8. Machine learning model: The app should use a pre-trained machine learning model to analyze the pictures and text inputs and determine the likelihood of depression.

IV. SYSTEM DIAGRAM

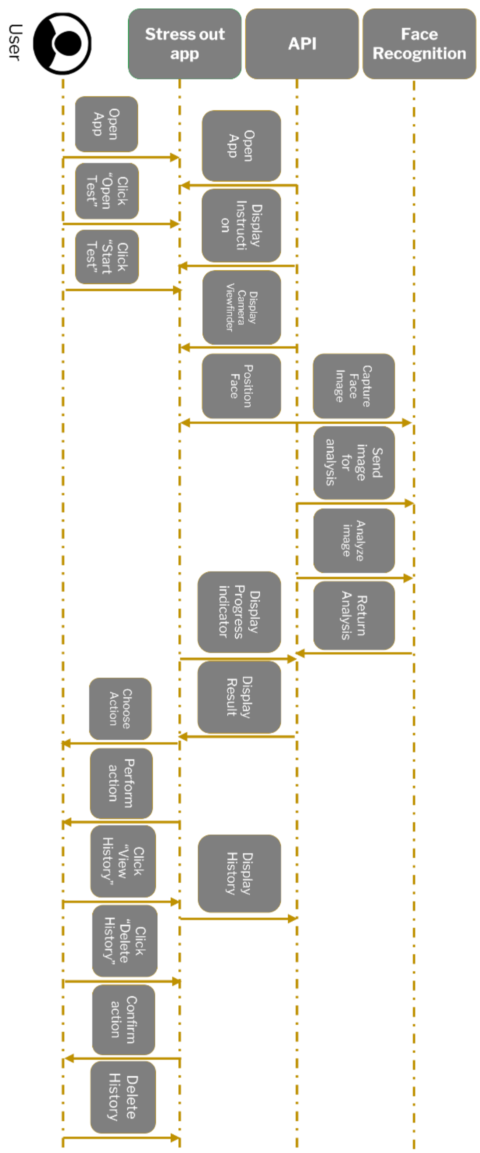

Figure 4.1 Sequence Diagram

Sequence Diagram shows interaction between user and StressOut app. It also demonstrates underlining components involved in the StressOut application work. The models are trained to work on 100x100 pixels picture. The application automatically crops the picture to the required size. To make sure that cutting out process won't cut necessary pieces of picture (cutting process might cut half of the face in any given picture) we create a special frame for user to put his face in. That will allow for constant picture quality with all necessary attributes present, without compromising user comfort (user won't have to take multiple pictures until model is satisfied with the result).

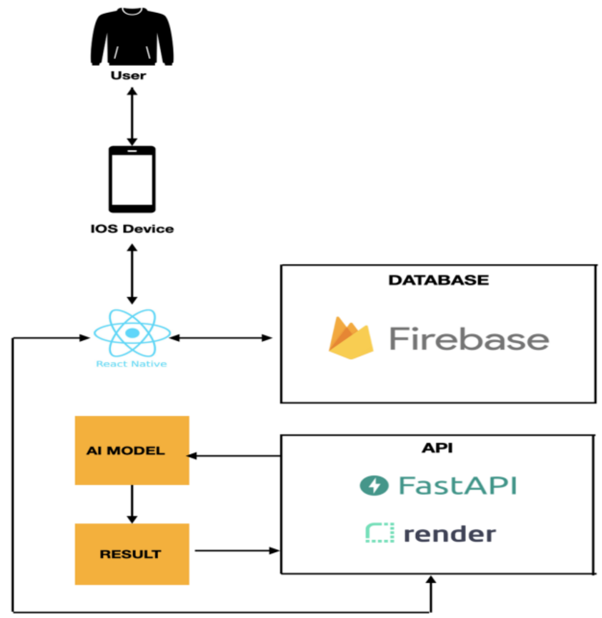

Figure 4.2 Conceptual Architecture Diagram

The Conceptual Architecture Diagram shown above represents high-level conceptual design of a system or software application. It provides an overview of the major components of the system and how they interact with each other to achieve the desired functionality. The mobile application sends an API request to the API server which is hosted on AWS. For API servers we are using node.js, which after the request has been processed then interacts with the Firebase database servers. Firebase provides authentication, storage and NOSQL database through firestorm.

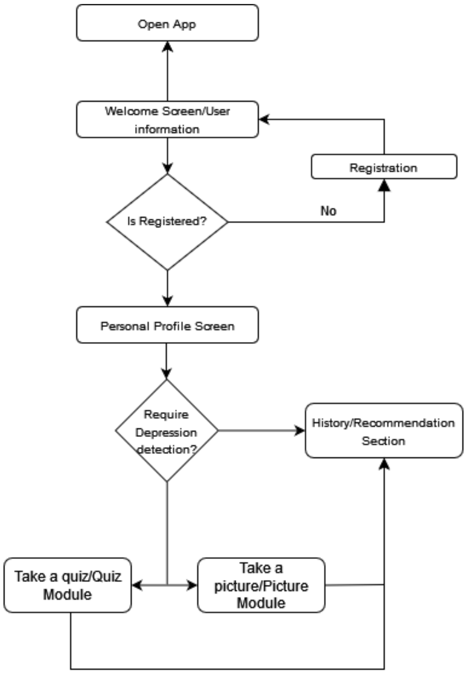

Figure 4.3 Control flow diagram

The current diagram shows control flow diagram shows sequential steps for StressApp users for successful of application. It is important to understand that taking a quiz will make predictions more accurate which is why the application is prompting users to take a quiz as well as taking pictures.

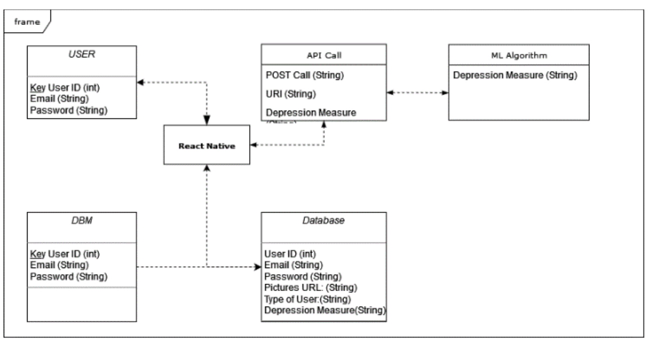

Figure 4.4 UML diagram

The diagram depicted in figure 4.4 is an UML diagram for StressApp. User has 3 main features: user id, email, and password. React native is center point for the application for communicating with ML algorithms through API calls, database and users. Database stores the main information about the user as well as pictures uploaded by the user.

V. MACHINE LEARNING MODEL

We developed a convolutional neural network (CNN) for our machine learning model, which comprises four convolutional layers, followed by two fully connected layers and an output layer. The input shape is (200, 200, 1), and the output class has two categories.

The convolutional layers have filter sizes of (3,3), with 128, 256, 512, and 512 filters respectively. The stride and padding are not explicitly specified, so they default to a stride of (1,1) and 'valid' padding. Each convolutional layer is followed by a max pooling layer with a pool size of (2,2) and a dropout layer with a rate of 0.4.

The two fully connected layers contain 512 and 256 neurons respectively, followed by dropout layers with rates of 0.4 and 0.3 respectively. The output layer has two neurons and uses the SoftMax activation function.

We trained the model using the Adam optimizer and the categorical cross-entropy loss function. To prevent overfitting, we used an early stopping callback that stops training if the validation loss does not improve for 10 epochs. The model's weights are initialized and updated automatically during training to minimize the loss function.

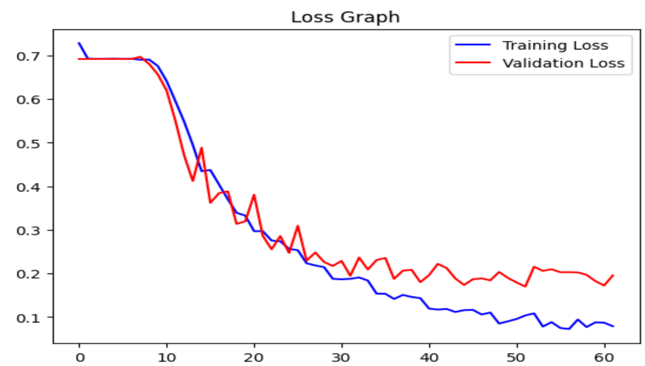

Figure 5.1 Loss Graph

In summary, our CNN model uses multiple convolutional and fully connected layers to classify input images into two categories with high accuracy. The model's architecture and training parameters are designed to prevent overfitting and maximize performance.

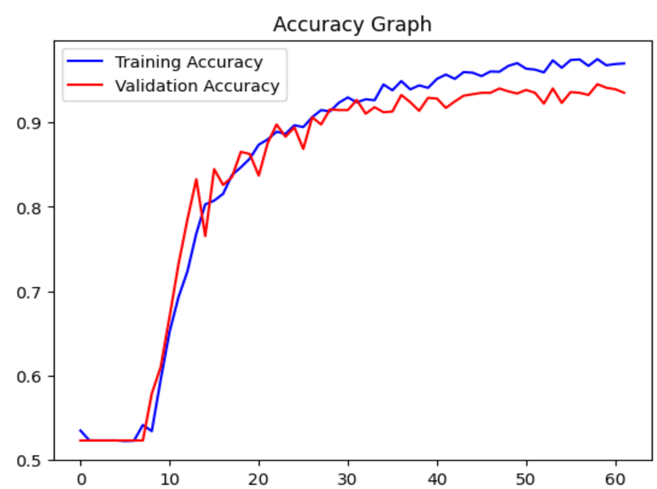

Figure 5.2 Accuracy Graph

VI. API DOCUMENTATION

The endpoint uses the Fast API library and can be accessed using HTTP GET or POST URL requests. The GET request simply returns a dictionary with a greeting message: "Hi there": "You reached the endpoint!". The POST request expects a base64-encoded image in the body of the request and passes it to the predit_depression() function.
The predit_depression() function takes the base64-encoded image as input, preprocesses it, and uses a TensorFlow model to predict the probability of depression. If the probability is less than 0.01, the function returns a “<0.01” prediction. Otherwise, it rounds the probability to two decimal places and returns it as a string.
The preprocess_image() function takes the base64-encoded image as input, decodes it, converts it to a grayscale image, detects faces in the image using the Haar Cascade classifier, crops and resizes the image to 100x100 pixels, normalizes it, and expands the dimensions of the image to match the input dimensions of the TensorFlow model.
The POST request should include a JSON object with the following properties:
Key: “image”, Value: base64-encoded image
200 OK: The API will return this response when the request was successful, and the app was able to analyze the image data provided by the user. The response will include a JSON object with the following properties:
Key: “Prediction”
Value: String – A score between 0 and 1 indicating the probability of depression
500 Internal Server Error: The API will return this response when there was an error on the server side, or the image data was corrupted.

VII. CONCLUSION

In conclusion, the StressOut app is a novel tool that utilizes machine learning to detect depression from picture and text inputs. The app is designed to be user-friendly and accessible, with a simple and intuitive interface that encourages regular use.

The app's architecture includes a convolutional neural network and natural language processing algorithms, which were trained on large datasets to accurately identify signs of depression. The app's requirements were carefully designed to ensure compatibility with iOS devices, while complying with data privacy regulations and ensuring that user data is kept secure.

The StressOut app has the potential to be a valuable resource for individuals struggling with depression, as it provides a convenient and discreet way to monitor their mental health. It may also serve as a useful tool for mental health professionals, as it can provide valuable insights into a patient's mental state and improve the accuracy of diagnoses.

While the StressOut app represents a significant step forward in the development of mental health technologies, there are still challenges that must be addressed. These include improving the accuracy and reliability of the machine learning algorithms, expanding the app's functionality to include additional mental health disorders, and ensuring that the app is accessible to individuals from diverse backgrounds and cultures.

Overall, the StressOut app represents an exciting development in the field of mental health and has the potential to make a significant impact on the lives of individuals struggling with depression.

VIII. BIBLIOGRAPHY

[1] He K. et al. Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – C. 770-778.

[2] L. Xu, X. Shu and J. Shu, "Research on Depression Tendency Detection Based on Image and Text Fusion," *2022 5th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, Chengdu, China, 2022, pp. 326-331, doi: 10.1109/ICAIBD55127.2022.9820295.

[3] A. Ashraf, T. S. Gunawan, F. D. A. Rahman, M. Kartiwi, N. Ismail and Ulfiah, "A Summarization of the Visual Depression Databases for Depression Detection," *2020 6th International Conference on Wireless and Telematics (ICWT)*, Yogyakarta, Indonesia, 2020, pp. 1-6, doi: 10.1109/ICWT50448.2020.9243625.

[4] A. M. Putri, K. Wijaya, O. A. Salomo, A. A. Santoso Gunawan and Anderies, "A Review Paper: Accuracy of Machine Learning for Depression Detection in Social Media," *2022 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT)*, Solo, Indonesia, 2022, pp. 39-45, doi: 10.1109/COMNETSAT56033.2022.9994553.