Grenze International Journal of Engineering and Technology, June Issue

Suicidal Tendency Detection using CNN Algorithm

Mr.B.V.Satish[1],V.Lakshmi Prathyusha[2], P. Naga Anupriya[3], N.Kasi Mounika[4]and M.Yoshitha[5]
[1-5]Prasad V. Potluri Siddhartha Institute of Technology/Information Technology, Vijayawada, India
Email: vsatish.phd@gmail.com, lakshmiprathyushaveturi@gmail.com, nagaanupriya999@gmail.com, kasimounika0l@gmail.com, yoshithamadala85564@gmail.com

***Abstract*— Some people have the intentions of killing themselves such cases are called Suicides. Detection of suicidal tendencies means the identification of an individual who has suicidal thoughts. Prevention of suicide can happen with early detection and they can be taken care of through healthcare professionals. This study mainly aims at the detection of the suicidal tendency of an individual using deep learning algorithms through non-image input.**

***Index Terms*— CNN, random forest, tkinter, numpy, sklearn, keras, pandas, lightgbm.**

I. INTRODUCTION

Suicidal tendency detection is necessary as many people nowadays tend to attempt suicide because of their job or stress factors or getting into any abnormal situations and so. This should be noticed and reduced by using the latest technology such as machine learning techniques, deep learning, etc. Our project mainly focused on the application of one of the deep learning algorithms i.e., the CNN algorithm to detect the suicidal tendency of an individual by having many input factors through a dataset collected. The non-image dataset passes through the layers of CNN and finally predicts whether a person tends of suicide or not. There is an already existing system of suicidal tendency detection of an individual using the machine learning algorithm i.e., Random Forest algorithm but it gives less accuracy and hence we had done using the deep learning algorithm i.e., CNN.

II. PROPOSED SYSTEM

The proposed system is detecting the suicidal tendency using a deep learning algorithm i.e., Convolutional Neural Network (CNN) algorithm. Generally, CNN takes the input with an image format; if not, machine translation must be done before the features are extracted. It gives a reliable outcome due to the correlation matrix it forms using the input. It is well suited for image and sequence data and gives better accuracy. CNN algorithm has faster training time and reduced human bias

III. TECHNOLOGIES USED

A. Anaconda Navigator- Jupyter Notebook:

It doesn't require an internet connection which runs on the local desktop. It is a free, open-source, interactive web tool. It can also be installed on a remote server and accessed through the internet.

Grenze ID: 01.GIJET.9.2.351

B. Python Libraries

The libraries of Python used here are numpy, pandas, keras, matplotlib, sklearn, and tkinter.

- The numpy is to operate the arrays and matrices, pandas are used to deal with the ML tasks and data analytics and also to load the data from the CSV file to the data frame of the panda.
- Pandas is an open-source data manipulation and analysis library. It proves data structures for storing and manipulating data and tools to read and write data in various formats, including CSV, Excel, and SQL databases.
- Matplotlib is used to have static, animated, and interactive visualizations for our data.
- Keras is an open-source deep-learning framework in Python. It is used to build and train deep-learning models. It supports neural network architectures.
- Scikit-learn (sklearn) is a machine learning library for Python that provides various tools for data pre-processing, model selection, and evaluation and for both supervised and unsupervised learning algorithms.
- Tkinter is a Python library that is used to create GUI (Graphical User Interface) applications. Tkinter makes developers create cross-platform desktop applications with ease.

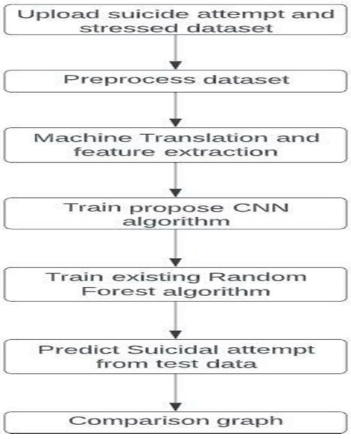

Figure 1: Architecture of Proposed System.

IV. TRAINING DATASET

A. CNN Algorithm:

CNN stands for Convolutional Neural Network. It is a type of deep neural network commonly used in image and video processing tasks. The layers present in CNN are convolutional, pooling layers, and fully connected layers.

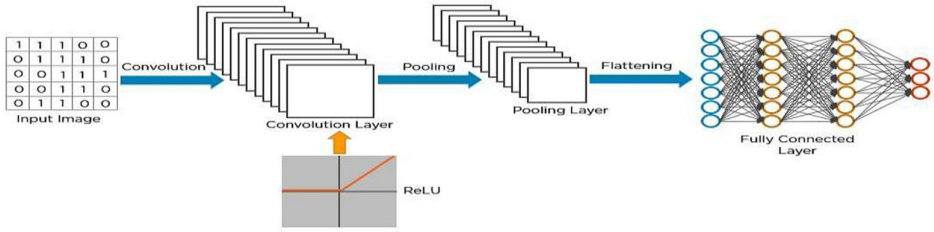

Figure 2: Layers in CNN Algorithm.

lt is a machine-learning algorithm that is used for both regression and classification. lt is an ensemble learning method that is used to build decision trees. The outputs of decision trees are combined to make predictions.

B. Random Forest Algorithm:

lt is a machine-learning algorithm that is used for both regression and classification. lt is an ensemble

learning method that is used to build decision trees. The outputs of decision trees are combined to make predictions

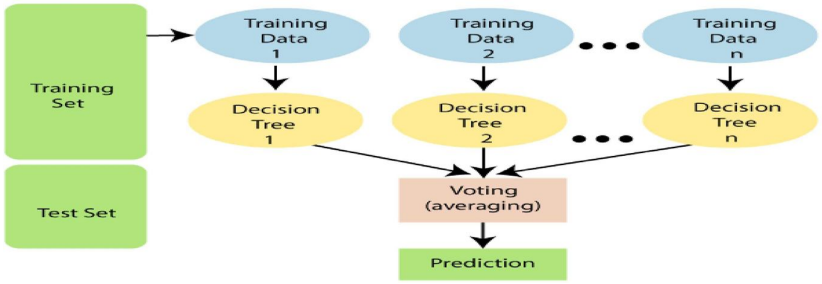

Figure 3: Steps in Random Forest Algorithm.

V. Simulations And Experimental Results

A. Machine Translation and Feature Extraction:

Machine translation in the CNN algorithm is the translation of text from one language to another using convolutional neural networks. The process of machine translation is encoding the input text using a sequence of convolutional and pooling layers which extracts the important and useful features. CNN algorithm is used to extract important features from the input (either image or non-image). The features are extracted using convolutional layers. These layers use filters that scan the entire input data and create a feature map and each filter is applied in the form of a sliding window approach. The output of the convolutional layer is passed through an activation function such as ReLU. Now, the output is flattened and this is passed through one or more fully connected layers. These are responsible to make final predictions.

```
In [11]: dataset
Out[11]:
```

	gender	sexuallity	age	race	bodyweight	virgin	prostitution_legal	pay_for_sex	friends
0	1	2	35	18	0	1	0	0	0.0
1	1	0	21	18	3	1	0	0	0.0
2	1	2	22	18	2	1	0	0	10.0
3	1	2	19	18	2	1	1	0	8.0
4	1	2	23	18	2	0	0	1	10.0
...	...	...	...	...	...	...	...	...	...
464	0	2	26	18	2	1	0	2	10.0
465	1	2	31	18	0	0	0	2	10.0
466	0	2	17	0	0	1	0	0	20.0
467	1	2	18	0	1	1	0	0	16.0
468	1	2	28	18	0	1	0	2	3.0

469 rows × 16 columns

Figure 4: Dataset that is used

Dataset after applying machine translation

	gender	sexuallity	age	race	bodyweight	virgin	prostitution_legal	\
0	1	2	35	18	0	1	0	
1	1	0	21	18	3	1	0	
2	1	2	22	18	2	1	0	
3	1	2	19	18	2	1	1	
4	1	2	23	18	2	0	0	
..	...	...	...	...	...	...	...	
464	0	2	26	18	2	1	0	
465	1	2	31	18	0	0	0	
466	0	2	17	0	0	1	0	
467	1	2	18	0	1	1	0	
468	1	2	28	18	0	1	0	

	pay_for_sex	friends	social_fear	stressed	what_help_from_others	\
0	0	0.0	1	1	37	
1	0	0.0	1	1	41	
2	0	10.0	1	1	5	
3	0	8.0	1	1	26	

Figure 4: Dataset after machine translation.

B. Training with the Proposed CNN Algorithm:

Firstly, the dataset is pre-processed by reshaping it to a 4D tensor and one-hot encoding the output layer. The CNN model is then defined using the Sequential API from Keras. This helps in the easy stacking of layers. The model is then compiled with an optimizer, loss function, and performance metric. The training process is done with the fit method. This trains the model on the input data and stores the training history. Finally, various performance metrics evaluate the model on a testing dataset. The final results are stored in corresponding arrays.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 16, 1, 32)         64

 max_pooling2d (MaxPooling2D  (None, 16, 1, 32)        0
 )

 conv2d_1 (Conv2D)           (None, 16, 1, 32)         1056

 max_pooling2d_1 (MaxPooling  (None, 16, 1, 32)        0
 2D)

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 256)               131328

 dense_1 (Dense)             (None, 2)                 514

=================================================================
Total params: 132,962
Trainable params: 132,962
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: The model is passing through the layers of CNN and displays the shape of each layer.

The epochs are used to find the accuracy of a model. It refers to the number of times the entire dataset is passed forward and backward through the neural network during training. We had given 70 as the batch size and thus 70 epochs had run. Finally, we got 94% accuracy for the model trained using the CNN algorithm.
The model is then compiled with an optimizer, loss function, and performance metric. The training process is done with the fit method. This trains the model on the input data and stores the training history.
Finally, various performance metrics evaluate the model on a testing dataset. The final results are stored in corresponding arrays.

```
None
Epoch 1/70
48/48 - 4s - loss: 1.3124 - accuracy: 0.5638 - 4s/epoch - 83ms/step
Epoch 2/70
48/48 - 0s - loss: 0.7533 - accuracy: 0.6159 - 238ms/epoch - 5ms/step
Epoch 3/70
48/48 - 0s - loss: 0.8276 - accuracy: 0.6549 - 291ms/epoch - 6ms/step
Epoch 4/70
48/48 - 1s - loss: 0.6993 - accuracy: 0.6758 - 540ms/epoch - 11ms/step
Epoch 5/70
48/48 - 0s - loss: 0.5916 - accuracy: 0.7109 - 417ms/epoch - 9ms/step
Epoch 6/70
48/48 - 0s - loss: 0.5503 - accuracy: 0.7266 - 497ms/epoch - 10ms/step
Epoch 7/70
48/48 - 0s - loss: 0.5964 - accuracy: 0.7083 - 402ms/epoch - 8ms/step
Epoch 8/70
48/48 - 0s - loss: 0.4933 - accuracy: 0.7682 - 412ms/epoch - 9ms/step
Epoch 9/70
48/48 - 0s - loss: 0.5728 - accuracy: 0.7253 - 352ms/epoch - 7ms/step
Epoch 10/70
48/48 - 0s - loss: 0.4437 - accuracy: 0.7891 - 487ms/epoch - 10ms/step
Epoch 11/70
48/48 - 0s - loss: 0.5403 - accuracy: 0.7318 - 424ms/epoch - 9ms/step
Epoch 12/70
48/48 - 0s - loss: 0.5205 - accuracy: 0.7487 - 395ms/epoch - 8ms/step
48/48 - 1s - loss: 0.1864 - accuracy: 0.9219 - 505ms/epoch - 11ms/step
Epoch 60/70
48/48 - 0s - loss: 0.1612 - accuracy: 0.9414 - 441ms/epoch - 9ms/step
Epoch 61/70
48/48 - 0s - loss: 0.1686 - accuracy: 0.9271 - 315ms/epoch - 7ms/step
Epoch 62/70
48/48 - 1s - loss: 0.1617 - accuracy: 0.9388 - 530ms/epoch - 11ms/step
Epoch 63/70
48/48 - 0s - loss: 0.1693 - accuracy: 0.9232 - 470ms/epoch - 10ms/step
Epoch 64/70
48/48 - 0s - loss: 0.1500 - accuracy: 0.9388 - 483ms/epoch - 10ms/step
Epoch 65/70
48/48 - 0s - loss: 0.1687 - accuracy: 0.9310 - 436ms/epoch - 9ms/step
Epoch 66/70
48/48 - 0s - loss: 0.1541 - accuracy: 0.9401 - 495ms/epoch - 10ms/step
Epoch 67/70
48/48 - 0s - loss: 0.1705 - accuracy: 0.9349 - 421ms/epoch - 9ms/step
Epoch 68/70
48/48 - 0s - loss: 0.1360 - accuracy: 0.9479 - 471ms/epoch - 10ms/step
Epoch 69/70
48/48 - 0s - loss: 0.1519 - accuracy: 0.9336 - 392ms/epoch - 8ms/step
Epoch 70/70
48/48 - 1s - loss: 0.1356 - accuracy: 0.9401 - 533ms/epoch - 11ms/step
5/5 [==============================] - 1s 10ms/step
```

Figure 7: Epochs for finding accuracy.

Below is the PERFORMANCE METRICS of using the CNN Algorithm:

Propose CNN Accuracy on Test Data : 94.8051948051948
Propose CNN Precision on Test Data : 95.29818496110632
Propose CNN GBM Recall on Test Data : 94.46801289665704
Propose CNN GBM FSCORE on Test Data : 94.73324213406293

Figure 8: Performance metrics of the model trained with the CNN algorithm.

C. Training with the existing Random Forest Algorithm:

All the steps from pre-processing the data to feature extraction are the same as using the CNN algorithm and later the training data is trained by initializing a random forest classifier and later on evaluating the performance metrics like accuracy, precision, recall, and Fl score.

Below is the PERFORMANCE METRIC of using the RANDOM FOREST Algorithm:

Existing Random Forest Accuracy on Test Data : 90.25974025974025
Existing Random Forest Precision on Test Data : 90.31713900134952
Existing Random Forest Recall on Test Data : 90.55659256745291
Existing Random Forest FSCORE on Test Data : 90.24946182094466

Figure 9: Pe1formance metrics of the model trained with the Random Forest algorithm.

D. Prediction of Suicidal Tendency:

The training dataset is used to train the model which contains a column "attempt_suicide", and has options as yes or no. This column doesn't present in the testing dataset. The testing dataset is loaded and the trained model is used to predict whether a person is having suicidal tendencies on the test data.

```
testData = testData.values

#It converts the testData DataFrame to a numpy array and reshapes it.
testData = testData.reshape(testData.shape[0],testData.shape[1],1,1)

#It uses the trained model classifier to predict the output for the test data.
predict = classifier.predict(testData)

#It converts the predicted output into a one-dimensional array using np.argmax().
predict = np.argmax(predict, axis=1)
#print(predict)

#t uses a for loop to display the results of the prediction for each row of the test data in the text widget.
#If the prediction is 1, it displays that a "SUICIDAL Depression" was detected;
#otherwise, it displays that "NO SUICIDAL Depression" was detected.
for i in range(len(predict)):
    if predict[i] == 1:
        text.insert(END,str(temp[i])+" ====> SUICIDAL TENDENCY DETECTED \n\n")
    if predict[i] == 0:
        text.insert(END,str(temp[i])+" ====> NO SUICIDAL TENDENCY DETECTED \n\n")
```

Figure I0: Code to predict suicidal tendency.

E. Comparison Graph between CNN and Random Forest Algorithm:

The dataset is trained with both the algorithms (CNN and Random Forest) and measured the following metrics after training with them.

i. Accuracy
ii. Recall
iii. Precision
iv. Fl Score

['5/19/2016 20:03:19' 'Female' 'Bisexual' 30 '$50,000 to $74,999'
'White non-Hispanic' 'Normal weight' 'Yes' 'No' "Yes but I haven't" 2.0
'No' 'Yes' "I don't want help, More general stuff" 'Employed for wages'
'Not telling' 'Bachelor's degree'
'Cosmetic survey, Joined a gym/go to the gym, Therapy, Other exercise, join clubs/socual clubs/meet ups, Fashion makeup personality etc'] ====> SUICIDAL TENDENCY DETECTED

['5/28/2016 17:04:03' 'Female' 'Straight' 25 '$0' 'White non-Hispanic'
'Normal weight' 'Yes' 'No' 'No' 2.0 'Yes' 'No' "I don't want help"
'A student' 'student teacher' 'Bachelor's degree' 'Other exercise'] ====> NO SUICIDAL TENDENCY DETECTED

['5/28/2016 20:36:53' 'Male' 'Straight' 22 '$1 to $10,000' 'Black'
'Normal weight' 'Yes' 'No' 'No' 15.0 'No' 'No' 'Set me up with a date'
'A student' 'Guest Advisor' 'Some college, no degree' 'None'] ====> NO SUICIDAL TENDENCY DETECTED

['5/28/2016 20:56:31' 'Male' 'Straight' 23 '$0' 'White non-Hispanic'
'Normal weight' 'Yes' 'No' 'No' 2.0 'Yes' 'No' "I don't want help"
'A student' '...' 'Some college, no degree' 'Therapy, Other exercise'] ====> NO SUICIDAL TENDENCY DETECTED

Figure I I: Suicidal tendency prediction on the testing dataset.

```
a = accuracy_score(y_test,predict)*100
p = precision_score(y_test, predict,average='macro') * 100
r = recall_score(y_test, predict,average='macro') * 100
f = f1_score(y_test, predict,average='macro') * 100

accuracy.append(a)
precision.append(p)
recall.append(r)
fscore.append(r)
```

Figure 12: Mathematical approach to calculate performance metrics.

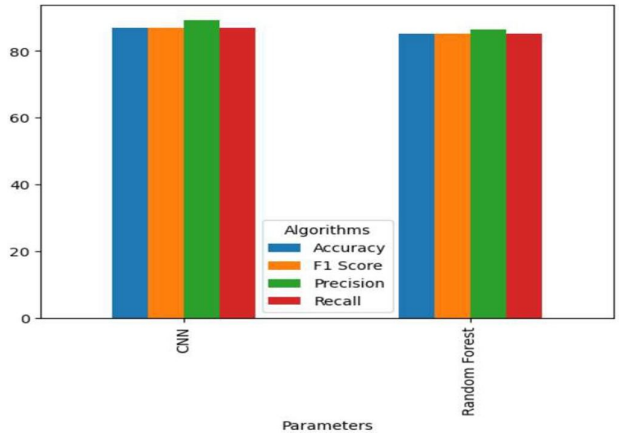

Figure 13: Comparison between CNN and Random Forest algorithm ·s performance metrics.

VI. WORKING FUNCTIONALITY

Step 1: Upload the training dataset.
Step 2: Pre-process the uploaded dataset to fill the null values if any with zero.
Step 3: Apply Machine Translation and Feature Extraction on the dataset that is pre-processed.
Step 4: Train the model with the proposed CNN algorithm and run the epochs to find its accuracy ofit.
Step 5: Now, train the model again with the existing Random Forest algorithm.
Step 6: Upload the testing dataset.

Step 8: A comparison graph is displayed between the accuracy of the CNN algorithm and the Random Forest algorithm.

VII. Future Scope

Suicide risk assessment may benefit from a combination of various types of data, including physiological signals, speech patterns, and social media activities. Integrating these data types may provide a more comprehensive understanding of an individual's emotional state, cognitive functioning, and behavioral patterns. Suicide risk is unique to each individual, and therefore, a personalized approach may be more effective in identifying those at risk. Personalized models may take into account an individual's history, socio- demographic factors, and other relevant information to provide tailored risk assessments.

VIII. Conclusion

To conclude, the use of a CNN algorithm for the detection of suicidal tendencies using non-image datasets is a relatively new and innovative approach. While CNNs are traditionally used for image classification, recent studies have shown promising results for using CNNs with non-image data as well. The use of a CNN for this task can potentially provide better accuracy and reduce the risk of false positives. The process involves converting the non-image data into a 2D image format to feed into the CNN. Pre-processing techniques such as normalization and feature scaling are applied before training the model on the dataset. The model can then be used to predict whether an individual is at risk of suicidal tendencies or not.
However, this approach is still in the early stages of development, and further research is needed to improve its accuracy and effectiveness. Nevertheless, the use of machine learning algorithms has the potential to provide valuable insights into mental health conditions and can help in the development of effective intervention strategies.

References

[1] Hayes, L.M., 2013. Suicide prevention in correctional facilities: Reflections and next steps. International Journal of Law and Psychiatry 36, 188-194.

[2] S. Lee et al., "Detection ofa Suicide by Hanging Based on a 3-D Image Analysis," in IEEE Sensors Journal, vol. 14, no. 9, pp. 2934-2935, Sept. 2014. doi: 10.1109/JSEN.2014.2332070.

[3] Calderon-Vilca, H. D., Wun-Rafael, W. I., & MirandaLoarte, R. (2017), "Simulation of suicide tendency by using machine learning", 2017 36th International Conference of the Chilean Computer Science Society (SCCC). doi:10.l 109/sccc.2017.8405128.

[4] Hu, Z., Hu, Y., Wu, B., & Liu, J. (2017), "Hand Pose Estimation with CNN-RNN", 2017 European Conference on Electrical Engineering and Computer Science (EECS). doi:10.1109/eecs.2017.91.