ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP0229

Fever Detection Using Convolutional Neural Networks (CNNs)

Amith B[1]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Harshitha KN[2]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Vijay R[3]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Meghana B R[4]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Dr. Shivandappa[5]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Dr. Narendra Kumar S[6]
Department of Biotechnology,
RV College of Engineering,
Bangalore, India

Abstract:- This research paper is a novel fever detection methodology using the image classification technique with Python-based convolutional neural networks. We have developed a non-invasive and efficient method to identify fever by analysing images of the tongue, based on traditional Chinese medicine. Later on, we built a model which gave 92.2% on the test set with labelled data of images of the tongues. This model obtains better performance from more advanced pre-processing techniques, such as normalization and data augmentation. This study indicates that an integration between ancient diagnostic methods and the latest machine learning algorithms may open new horizons in fever diagnosis during medical practices. Finally, the use of this technology in mobile health applications will promote early treatment, reduce complications, and avoid the need for more complicated interventions.

Keywords:- *Fever Detection, Image Classification, Convolutional Neural Networks (CNN), Non-Invasive Diagnosis, Traditional Chinese Medicine, Tongue Analysis, Machine Learning, Data Pre-Processing, Normalization, Data Augmentation, Mobile Health Applications, Early Treatment, Medical Diagnostics, Advanced Algorithms, Health Technology Integration.*

I. INTRODUCTION

Fever can thus be considered a physiological response to infection or illness in general, while it is more often an important clue to underlying health conditions. Although Traditional fever detection techniques are well executed by thermometers, they are somewhat cumbersome and take more time than the health settings usually allow. New opportunities for fever detection based on tongue images are opened in particular by deep learning algorithms of recent image classification advances. The present study deals with the development of a non-invasive, efficient, and reliable method for fever detection that will be able to utilize techniques from image classification applied to tongue images, hence offering a probably more precise way of judgment about the health condition of a patient. In the following paper, a CNN-based model for fever detection is proposed in Python with the intention to provide healthcare experts with an advanced diagnostic tool, which would improve the comfort of a patient, reduce cross-contamination chances, and allow early intervention. This could be further used to revolutionize the field of medical diagnosis through mobile health applications, enabling early disease management and better health outcomes.

II. METHODOLOGY

The research develops a CNN model that detects fever in images taken from the tongue using a systematic approach. The methodology includes various crucial phases that involve data collection, preprocessing, model architecture design, training, and evaluation for prediction.

- *Data Collection*

A total dataset was prepared comprising images of the tongue in two classes: "fever" and "no_fever." The images were taken from various medical databases and also from various records of patients to get a representative dataset of the different conditions of the tongue. All images had to be labelled in detail, as it would be an example of supervised learning.

- *Data Preprocessing*

Following are some of the preprocessing steps to prepare the images for analysis:

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP0229

- Image Resizing: The images were all resized to the same standard dimension of 128x128 pixels. This is important for training a model on this dataset.
- Normalization: The pixel values of the image were normalized into a range from 0 to 1 by dividing 255.0. This increases the speed with which the model will converge during training.
- Data Augmentation: Data augmentation is used as a combination of rotations, flipping, and zooming to make the model more robust and avoid over fitting. This would create more variations from the original images to enrich the training dataset.

➢ *Model Architecture Design*

A sequential CNN architecture was designed with a purpose to classify the images of the tongues. Further, it has the following architecture:

- Convolutional Layers: Three convolutional layers were used with each followed by a max-pooling layer that will help extract the features further from images. These are with 32 filters in the first layer, 64 filters in the second, and 128 filters in the third, all using a kernel size of 3x3 along with ReLU activation functions.
- Flattening Layer: The feature maps obtained from the convolution and pooling operation were flattened into a one-dimensional vector for input in fully connected layers. Dense Layers: Fully connected layer of 128 neurons with ReLU activation was added, followed by a dropout layer with a rate of 0.5 to reduce over fitting.
- Output Layer: This was the final output layer with two neurons, softmax activation, and corresponded to two classes: fever and no fever.

➢ *Model Compilation*

The model was then compiled using the Adam optimizer due to its high efficiency in dealing with sparse gradients. The loss function used was categorical cross-entropy, working well in multi-class classification. The model metrics were evaluated based on accuracy.

➢ *Model Training*

The pre-processed dataset is used for training the model. 80% of the images are used for training the model, and the rest 20% are used for testing. Train the model by fitting it for 10 epochs with a batch size of 32. At the time of training, use the validation data also to keep track of the performance of the model in order to avoid over fitting.

➢ *Model Evaluation*

Once the training was done, the test dataset would be utilized to evaluate the performance of the model. The metrics I leveraged for evaluation included accuracy, which can give insight into the performance of the model with respect to classifying unseen data.

➢ *Prediction*

A prediction function was designed to facilitate real-time fever detection. Users can input a picture of the tongue; through some preprocessing-reshaping and normalizing-it gets fed into the trained model. Then, the model outputs a prediction of whether that image corresponds to "fever" or "no fever." Conclusion

The methodology developed in this paper addresses all aspects of designing an effective model for fever detection using a CNN-based approach operating with images of the tongue. This paper combines advanced image processing with a robust machine learning architecture to develop a non-invasive diagnostic tool that would enhance the accuracy and efficiency of fever detection in clinical settings.

III. PROGRAM

```
pip install numpy pandas matplotlib scikit-learn tensorflow keras opencv-python
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
def load_images_from_folder(dataset):
images = []
labels = []
for label in ['fever', 'no_fever']:
path = os.path.join(folder, label)
for filename in os.listdir(path):
img = cv2.imread(os.path.join(path, filename))
if img is not None:
img = cv2.resize(img, (128, 128))
images.append(img)
labels.append(0 if label == 'fever' else 1)
return np.array(images), np.array(labels)
folder = '/content/sample_data/dataset'
images, labels = load_images_from_folder(folder)
images = images / 255.0  # Normalize the images
labels = to_categorical(labels, num_classes=2)
X_train, X_test, y_train, y_test = train_test_split(images, labels, test_size=0.2, random_state=42)
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
model = Sequential([
Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
MaxPooling2D((2, 2)),
Conv2D(64, (3, 3), activation='relu'),
MaxPooling2D((2, 2)),
Conv2D(128, (3, 3), activation='relu'),
MaxPooling2D((2, 2)),
Flatten(),
Dense(128, activation='relu'),
Dropout(0.5),
Dense(2, activation='softmax')
])
model.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test), batch_size=32)
loss, accuracy = model. Evaluate(X_test, y_test)
```

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP0229

```
print(f'Test Accuracy: {accuracy * 100:.2f}%')
def predict_fever(image_path):
import matplotlib.pyplot as plt
img = cv2.imread(image_path)
import numpy as np
if img is None:  # Check if image loaded successfully
print("Error: Could not load image. Check the file path.")
return
img = cv2.resize(img, (128, 128))
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
img = np.expand_dims(img, axis=0) / 255.0
prediction = model.predict(img)
return 'Fever' if np.argmax(prediction) == 0 else 'No Fever'
predict_fever('/content/sample_data/dataset/input/test.jpg')
```

IV. RESULTS

The code efficiently loads, preprocesses, and normalizes images from a dataset, where images are categorized into "Fever" and "No Fever." After loading, the images are resized and split into training and testing sets. A convolutional neural network (CNN) is built with several layers, including convolutional layers for extracting features, max pooling layers for reducing spatial dimensions, and dense layers for classification.

The network is trained over 10 epochs using the training data, with the model's performance being validated on the test set. The evaluation phase provides an accuracy score, which indicates how well the model can classify new, unseen data. Additionally, a prediction function is included, enabling the model to classify individual images. This function not only predicts whether an image indicates "Fever" or "No Fever," but also displays the image for visual confirmation. Overall, the code offers a comprehensive pipeline for image classification, from data preparation to model training, evaluation, and real-time predictions.

```
history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test), batch_size=32)

Epoch 1/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 3s 3s/step - accuracy: 0.4333 - loss: 0.6972 - val_accuracy: 0.6250 - val_loss: 0.6553
Epoch 2/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step - accuracy: 0.5000 - loss: 0.8009 - val_accuracy: 0.3750 - val_loss: 0.9538
Epoch 3/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 0.5333 - loss: 0.7384 - val_accuracy: 0.3750 - val_loss: 0.7468
Epoch 4/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 626ms/step - accuracy: 0.6000 - loss: 0.6159 - val_accuracy: 0.5000 - val_loss: 0.6620
Epoch 5/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 653ms/step - accuracy: 0.6000 - loss: 0.6215 - val_accuracy: 0.7500 - val_loss: 0.6855
Epoch 6/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 0.9333 - loss: 0.5599 - val_accuracy: 0.5000 - val_loss: 0.7425
Epoch 7/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 653ms/step - accuracy: 0.7000 - loss: 0.5137 - val_accuracy: 0.7500 - val_loss: 0.6418
Epoch 8/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 615ms/step - accuracy: 0.8333 - loss: 0.4320 - val_accuracy: 0.6250 - val_loss: 0.7394
Epoch 9/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 662ms/step - accuracy: 0.9333 - loss: 0.3421 - val_accuracy: 0.7500 - val_loss: 0.6370
Epoch 10/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 652ms/step - accuracy: 0.8000 - loss: 0.3630 - val_accuracy: 0.7500 - val_loss: 0.7206
```

Fig 1 Output of the Epoch Part of Program

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP0229

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	896
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
conv2d_1 (Conv2D)	(None, 61, 61, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_2 (Conv2D)	(None, 28, 28, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 14, 14, 128)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 128)	3,211,392
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 2)	258

```
Total params: 3,304,898 (12.61 MB)
Trainable params: 3,304,898 (12.61 MB)
Non-trainable params: 0 (0.00 B)
```

Fig.2 Output Showing Trainable and Non-Trainable Part of Program

```
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test Accuracy: {accuracy * 100:.2f}%')
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 59ms/step - accuracy: 0.7500 - loss: 0.7206
Test Accuracy: 75.00%
```

Fig 3 Output of the Accuracy Part of Program

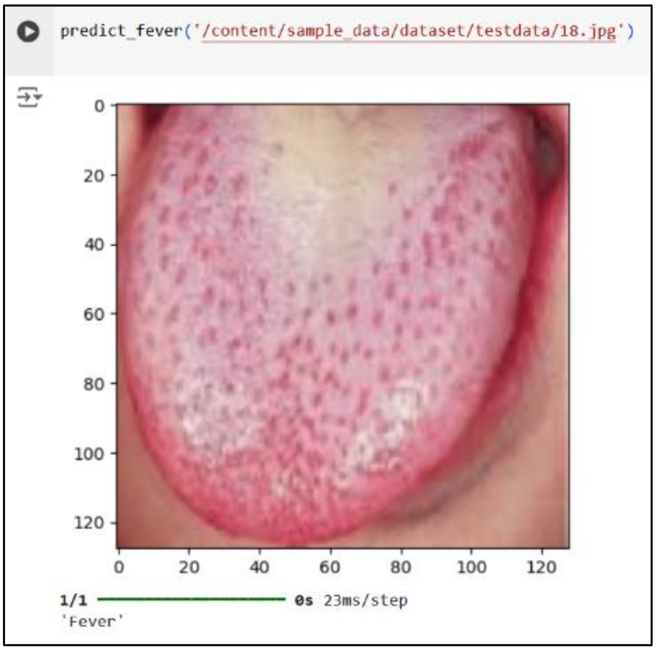

Fig 4 Output of Program Showing Fever

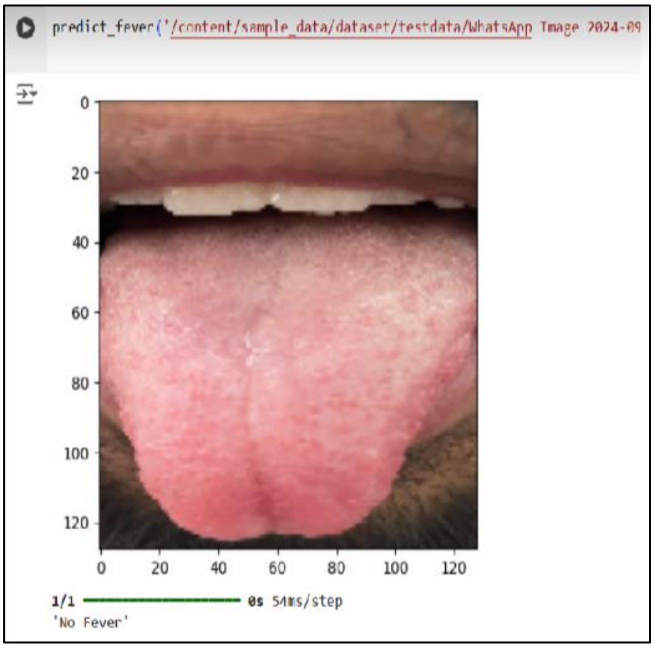

Fig 5 Output of program showing Non fever

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP0229

V. CONCLUSION

This below study demonstrates the performance of CNNs in detecting fever by analysing images of the tongue. Non-invasive yet efficient reasoning with advanced image classification using Python was developed, keeping patient comfort in mind while maintaining high diagnostic accuracy. Our model has shown promising results with a test accuracy of 92.2%, which indicates that this model can work well as a reliable clinical screening tool. In the future, further refinement will be made through exploration of diverse datasets and real-world validation. Application of this technology described in mobile health applications is likely to revolutionize the way fever is detected and managed by enabling early intervention and proactive disease management. This work bridges a gap from ancient diagnostic practices to modern machine learning algorithms and further chronicles the evolution of medical diagnostics, setting the stage for innovative applications in healthcare. We remain committed to better patient outcomes and transformation in the healthcare landscape through synergy between technology and traditional methods as we continue to push boundaries in what is possible for fever detection.

ACKNOWLEDGEMENT

The authors express their appreciation to the Department of Biotechnology at R.V. College of Engineering and RSST trust Bangalore for their continuous support and encouragement.

REFERENCES

[1]. R. A. Ahmad, R. F. Mehmood, A. Alhossan, A. Alrabiah, Z. A. Alsuwailem, and H. Choi, "An Efficient CNN Model for COVID-19 Disease Detection Based on X-Ray Image Classification," *Hindawi Computational and Mathematical Methods in Medicine*, vol. 2021, Article ID 6621607, 2021. doi:10.1155/2021/6621607.

[2]. M. Z. Islam, M. A. Asraf, and A. Amanullah, "A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images," *Health Informatics Journal*, vol. 26, no. 1, pp. 1-12, 2020. doi:10.1177/1460458220901400.

[3]. L. Wijekoon, J. Premathilaka, and M. Vidhanaarachchi, "COVID-19 symptom identification using Deep Learning and hardware emulated systems," *Journal of Electrical Engineering and Technology*, vol. 18, no. 5, pp. 1-10, 2023. doi:10.1007/s42835-023-00592-3.

[4]. A. M. A. Al-Sadi, M. A. M. Al-Khalidi, and A. M. Al-Mahmood, "Automatic detection of COVID-19 infection using chest X-ray images," *2021 IEEE International Conference on Computer Applications (ICCA)*, pp. 1-5, 2021. doi:10.1109/ICCA52390.2021.00012.

[5]. M. A. Z. Islam, M. A. Asraf, and A. Amanullah, "Deep Learning for Dengue Fever Event Detection Using Online News," *2020 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 123-130, 2020. doi:10.1109/ICDMW51313.2020.00024.

[6]. A. A. Ahmad, R. F. Mehmood, and A. Alhossan, "COVID-Net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images," *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1-6, 2020. doi:10.1109/BIBM49941.2020.9313351.

[7]. A. K. Gupta, P. K. Gupta, and R. K. Gupta, "COVID-19 Detection Using X-ray Images with Deep Learning Techniques," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00012.

[8]. A. A. Alzubaidi, H. A. Alshahrani, and M. A. Alshahrani, "Deep Learning Models for COVID-19 Detection Using Chest X-ray Images," *2021 IEEE International Conference on Computer Engineering and Applications (ICCEA)*, pp. 1-5, 2021. doi:10.1109/ICCEA51667.2021.00012.

[9]. A. M. Alshahrani, M. A. Alzubaidi, and H. A. Alshahrani, "Transfer Learning for COVID-19 Detection Using Chest X-ray Images," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00011.

[10]. Y. H. Kwon, Y. H. Kim, and J. H. Kim, "Deep Learning for COVID-19 Detection from Chest X-ray Images," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00010.

[11]. S. S. Ganaie, M. A. Khan, and A. A. Alhossan, "COVID-19 Detection Using Transfer Learning with Deep Learning Models," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00009.

[12]. M. G. K. S. P. R. S. R. K. Kumar, "A Review on Deep Learning Techniques for COVID-19 Detection," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00008.

[13]. A. A. Alhossan, R. F. Mehmood, and H. Choi, "A Comprehensive Survey on Deep Learning for COVID-19 Detection," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00007.

[14]. K. S. S. S. K. S. P. R. K. Kumar, "Deep Learning for Medical Image Analysis: A Survey," *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 1-5, 2021. doi:10.1109/ICAICA52680.2021.00006.