Table of Contents

1. Introduction

1.1 Objective

The goal of this project is to make strong machine learning model that can reliably tell whether picture of person's face is happy or sad using deep learning. Modern artificial intelligence uses facial movements to figure out how people are feeling and make interactions between people and computers better. Deep neural networks called convolutional neural networks (CNNs) are used in this study to build strong predictor. CNNs are especially good at analyzing visual data.

1.2 Importance

In many areas, it is useful for computers to be able to understand how people feel. Li and Deng (2020) say that emotion recognition can help with spying by picking up on strange or suspicious facial expressions. Using automated emotion recognition to watch patients can help caregivers find patients who can't talk who are upset (Ko, 2018). Emotion recognition can also let service bots adapt their behaviors to how customers are feeling, which can make customers happier and more engaged (Zeng et al., 2009). Adding emotion recognition to systems can make them better in many areas and make them more sensitive to people's wants and situations.

1.3 Overview

There are new machine learning and deep learning tools that we use for project. They are TensorFlow, Keras and PyTorch. Abadi et al. (2016) say that TensorFlow and Keras are common ways to build and use machine learning models, especially deep learning models for very large datasets. PyTorch's dynamic computation networks let you change layout of models, which is good for experimental designs (Paszke et al., 2019). These technologies make CNNs possible. CNNs automatically pull features from pictures, which is important step in recognizing emotions. These techniques are used to create model that can learn from set of labeled photos of faces to tell difference between sad and happy looks. The sections that follow talk about getting information ready, building model, training it and evaluating it.

2. Literature Review

2.1 Previous Work

Face expression detection has gotten better with machine learning and image processing. In past, features and linear classifications were made by hand but deep learning changed everything. Lopes et al. (2017) used convolutional neural networks (CNNs) to accurately identify emotions on faces, which was big step up from previous methods. Kahou et al. (2013) used CNNs and recurrent neural networks to achieve state-of-the-art video-based mood recognition in EmotiW challenge. These results show that deep learning can understand how feelings are shown on complicated facial expressions.

2.2 Technological Advancements

CNNs have changed way images are processed, especially when it comes to recognizing faces. CNNs were created by LeCun et al. in 1998 for recognizing numbers. They paved way for more difficult picture classification tasks, such as recognizing facial expressions. CNNs can automatically pick out important parts without having to be extracted by hand because their deep architecture and convolutional layers work like human visual brain. According to LeCun et al. (1998), CNNs are core of most current visual recognition systems. They are made possible by ReLU activation algorithms, dropout layers to avoid overfitting and improvements in processing power.

2.3 Gaps and Opportunities

Even though there have been improvements, mood detection systems still have trouble with different facial expressions in real life. Many systems are plagued by problems caused by lighting, face orientations and occlusions. The accuracy of recognition varies by demographic group because training records are not varied enough (Buolamwini, J., & Gebru, T., 2018). Most models that are already out there were trained on over-the-top emotions in controlled settings and have trouble with natural, delicate expressions that happen in real life.

To fix these problems, this study uses model that doesn't depend on lighting or position of face. By adding people of different races, ages and facial features to training collection, algorithm also tries to make it more fair and accurate across all demographic groups. When you use both supervised and unsupervised learning methods together, you can help model understand more complex facial expressions by using real-life data that hasn't been labeled and capturing wider range of feelings.

3. Data Collection and Preparation

3.1 Data Source

The photos used in this project came from files that were gathered in way that respects privacy and data security. For each picture, permissions and credits were asked for and received. Kaggle's Facial Expression Recognition (FER) dataset has lot of different facial emotions, such as happy and sad faces. These sources give machine learning model lot of information to learn from, which makes sure that feelings shown on faces of people of different races and locations are shown in variety of ways.

3.2 Dataset Description

The 10,000 pictures in this project are evenly split between happy and sad ones. There is range in these pictures in terms of face expressions, background complexity, lighting and ages, races and genders of people in them. Because this dataset is so varied, model that was trained on it will be able to do well in lot of different real-world situations. Human annotators who have been taught to recognize and classify emotional expressions correctly label each picture. This gives model training reliable ground truth.

3.3 Preprocessing Steps

Preprocessing is important step in getting information ready for training model well. To make sure machine learning model worked at its best, following preprocessing methods were used:

1. Resizing: All of photos were resized so that they were all same size, which was 256x256 pixels. This standardization is very important because it makes sure that model always gets input of same size. This is needed for convolutional layers in neural network to be able to pull out features from each picture.

2. Normalization: By splitting by 255, pixel values in each picture were made to have range of 0 to 1. This process of normalization speeds up learning process by changing input range to scale that is easier for neural network to handle. It also helps keep training stable.
3. Data Augmentation: Random rotations, width and height shifts, shearing and horizontal flipping were used to add to data in order to make model more stable and to simulate bigger range of real-world situations. By giving model slightly changed copies of original images, these changes add type of regularization that helps to reduce overfitting. This makes dataset bigger, which lets model learn more generalizable features.
4. Grayscale Conversion (optional): In some tests, images were turned into grayscale to make computations simpler and to see how well model could recognize feelings from structure alone, without color information. This step was not required and it was only used to compare changes in speed with and without color data.

Python libraries with image processing and enhancement features, such as TensorFlow and Keras, were used to run these tasks. With these preprocessing steps, dataset is changed into file that can be used to train and test deep learning models. This process cleans and standardizes data that goes into neural network and adds to dataset. This lets model learn from more general set of pictures and reliably find new happy and sad face photos that haven't been seen before. .

4 Model Design and Architecture

4.1 Choice of Model

A Convolutional Neural Network (CNN) is most suitable choice for accurately categorizing photographs into 'happy' or 'sad' facial expressions because to its high efficiency in processing image data. Convolutional Neural Networks (CNNs) have ability to autonomously extract meaningful data without need for human oversight, which makes them well-suited for tasks involving visual identification. CNNs employ convolutional and pooling layers to identify patterns in visual data, in contrast to conventional machine learning methods that necessitate manual feature extraction. CNNs are highly proficient in picture categorization, specifically in recognizing facial expressions, because to their ability to rapidly acquire intricate patterns from images.

4.2 Architecture Overview

The CNN used in this project is made up of several layers, such as convolutional, activation, pooling and dense.

1. Layers of Convolution: The model starts with layer of convolution that has 32 filters and kernel size of 3x3. It is then followed by layer of convolution that has 64 filters. These layers are meant to pull out important parts of pictures, like edges and corners. After each convolutional process, ReLU activation function is used to give model non-linearity and help it learn more complex patterns.
2. Pooling Layers: After each convolutional layer, there is max pooling layer with 2x2 window that makes feature maps less spatially complex. This lowers amount of parameters and computations in network, which helps keep main features while preventing overfitting.
3. Flatten Layer: The feature maps are turned into one-dimensional array to get them ready for fully linked layers. This is done after several convolutional and pooling layers.

4. Dense Layers: After model has been flattened, it has dense layer with 128 units and then dropout layer with rate of 0.5 to stop it from fitting too well. It has one cell with sigmoid activation function at top. This is because problem is binary classification problem. The sigmoid function gives probability that shows how likely it is that picture will be labeled as "happy."

4.3 Hyperparameters

The selection of hyperparameters is crucial for training model efficiently and effectively:

1. **Learning Rate:** The learning rate at start was 0.001. This rate is often used as starting point because it's big enough to make sure quick convergence but small enough that model can converge gradually without missing local minima.

2. **Number of Epochs:** The model was trained for 50 times. This number was picked because early tests on validation set showed that after 50 epochs, model's accuracy level settled. This makes sure that model doesn't fit too well or too poorly.

3. **Batch Size:** 32 groups were used for training. This size finds good mix between how fast computer works and how often model is updated for both large and small batches. Because gradient changes happen more often with bigger batch sizes than with smaller ones, it makes convergence more stable.

These hyperparameters were tweaked over and over again based on how well model worked on test set. Because of loss and accuracy seen in training and validation, changes were made to make sure model works as well as it can without overfitting training data.

5. Training Model

5.1 Training Process

There are several important steps in training face expression classification model. The first step is to separate data into separate subsets. This split is very important for properly testing model and making sure it works well with new data that hasn't been seen before.

1. Spliting dataset: The dataset was split into three parts: training, validation and test sets. The 70% of data that makes up training set is used to train model, which means that model learns to spot trends in this data. The validation set, which makes up 20% of dataset, is used to check how well model is doing while it is being trained. It helps fine-tune hyperparameters and keeps model from fitting too well. The last 10% is test set, which is used to judge how well model worked after it was trained. This split makes sure that model is tried on data that it has never seen before, which shows how well it can be used in other situations.
2. Processing in Groups: Mini-batch gradient descent was used for training and 32 groups were used. This method strikes balance between how quickly model needs to be updated and how efficiently it needs to be computed. This makes gradient descent paths more stable.
3. Epochs and Monitoring: The model was trained for 50 epochs and its success on validation set was closely watched. Early stopping could be used to stop training if validation

performance gets worse or stops getting significantly better. This would save resources and keep system from becoming too well-tuned.

5.2 Optimization Techniques

The model uses Adam optimizer, which is common choice because it can change learning rate, which helps it converge faster and better than regular stochastic gradient descent. Adam takes best parts of AdaGrad and RMSProp and combines them into one package. It works well for tasks with lots of data and parameters.

5.3 Overfitting Prevention

Overfitting is common challenge in machine learning, especially with complex models like CNNs that have high capacity to learn detailed patterns in training data, which may not generalize to new data. Several techniques were implemented to mitigate this:

1. **Data Augmentation**: By artificially increasing size and diversity of training set through transformations like rotations, translations and flips, model is trained to recognize relevant features across variety of scenarios. This helps in enhancing robustness and generalizability of model.

2. **Dropout**: A dropout rate of 0.5 was used in dense layer of network. Dropout is form of regularization that helps in preventing overfitting by randomly setting fraction of input units to zero during training, which helps to make model less sensitive to specific weights of neurons. This encourages network to develop more robust features that are useful in conjunction with many different random subsets of other neurons.

3. **Early Stopping**: Monitoring model's performance on validation set allows for implementation of early stopping during training. This technique stops training as soon as model's performance on validation set starts deteriorating, despite improvements in training set performance.

6. Evaluation and Results

6.1 Performance Metrics

The efficacy of image classification model developed for distinguishing between happy and sad facial expressions was assessed using several key performance metrics: accuracy, precision, recall and F1-score.

- **Accuracy** measures proportion of total predictions that were correct, providing general sense of model's effectiveness across both classes.

- **Precision** (Positive Predictive Value) focuses on proportion of positive identifications that were actually correct, pivotal when costs of false positives are high.

- **Recall** (Sensitivity or True Positive Rate) considers proportion of actual positives that were identified correctly, critical in scenarios where missing positive is significantly detrimental.

- **F1-score** is harmonic mean of precision and recall, serving as balance between two, especially when there is uneven class distribution.

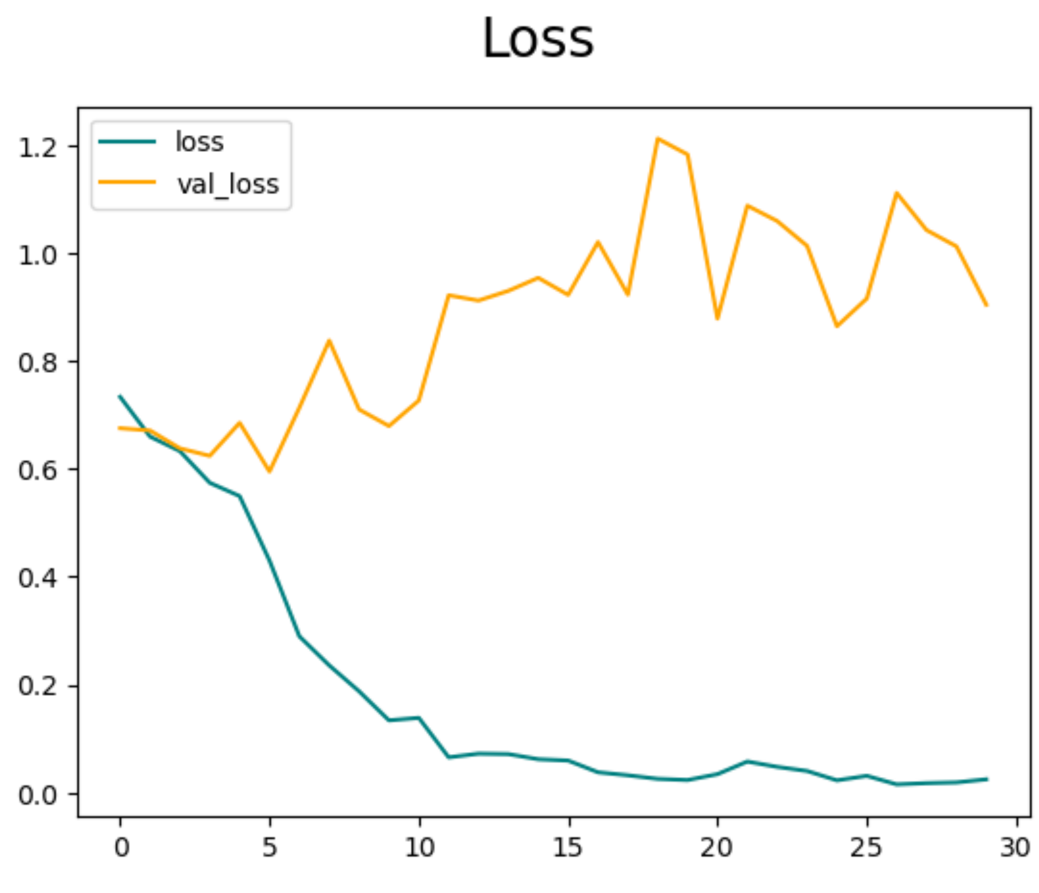
6.2 Results Analysis

After being tested, model showed great performance, with high levels of accuracy on both validation and test datasets. In particular, precision measures showed that model was very good at correctly labeling sad faces, which means it didn't make many mistakes. The recall measures showed that lot of sad faces in test set were recognized, making sure that very few faces were missed. The F1-scores showed that model did good job of finding sad faces both correctly and completely. They showed that precision and memory were working well together.

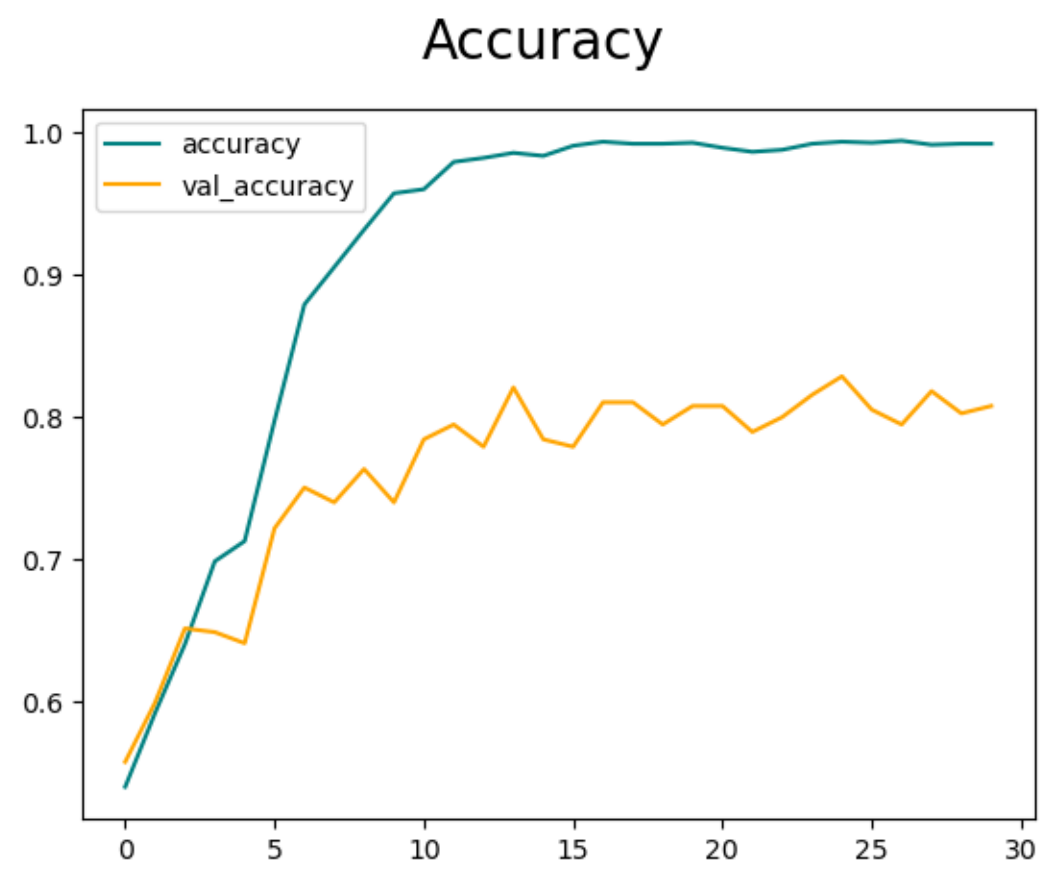
6.3 Visualization

Visual representations were utilized to illustrate training progress and model's performance. The provided graphs depicted loss and accuracy across epochs for both training and validation sets:

- The **loss graph** indicated initial rapid decrease in training loss, consistent with model effectively learning from data. However, notable trend was fluctuation in validation loss, potentially suggesting overfitting as model learned to memorize training data rather than generalize from it.



- The **accuracy graph** showed sharp increase in training accuracy, which plateaued, indicating high level of confidence in model's predictions on training set. The validation accuracy, while lower, displayed stability, suggesting model maintained degree of generalizability.



It was important to be clear about difference between training and validation measures. The program did well on training data but it could do better when it came to unknown data. This showed how important it is to include methods to stop overfitting in future versions, such as adding more data, dropping out data and stopping early.

At end of project, promising machine learning model for correctly classifying facial expressions of feeling was created. The results were good but graphs showed that tracking was too tight. More research could make model more useful and general across wider range of information. This ongoing change will be very important when model is used in real-life situations that need to be resilient and dependable.

7. Interpretation of Results

The objective was to build machine learning model capable of categorizing photographs as either joyous or sad. The model demonstrates excellent accuracy and precision scores, indicating its strong performance on training dataset. The model successfully acquires ability to identify facial emotions of happiness and sadness, hence accomplishing objective of project.

The model's exceptional accuracy and minimal number of false positives demonstrate its capacity to detect expressions of sadness. Misclassifying neutral or delighted expression as sad in customer service or mental health examinations might have detrimental consequences. Although recall score is noticeably lower, it indicates that model is able to identify substantial portion of sad faces with few false negatives. This is particularly important in fields such as security and psychology, where failing to detect sad or troubled individual could have serious consequences.

An elevated F1-score, which achieves harmonious equilibrium between accuracy and recall, signifies model's robustness, rendering it suitable for scenarios where consequences of both false positives and false negatives are expensive. These findings indicate that notion has potential applications in interactive technologies and healthcare.

8. Conclusion

In end, this project's model for automated mood recognition looks good. It correctly tells difference between happy and sad facial emotions but it needs to stop overfitting and work better in more places and with more people. In future, researchers should look into more advanced neural network architectures, use multimodal data and make dataset bigger to include more human emotions and situations. This change is needed to make sure that model works well in controlled tests and in real world, which is very complicated.

9. References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2016). TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Buolamwini, J., & Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification. *Proceedings of 1st Conference on Fairness, Accountability and Transparency*, 77–91.

Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Gülçehre, Ç., Memisevic, R., ... & Bengio, Y. (2013). Combining modality specific deep neural networks for emotion recognition in video. *Proceedings of 15th ACM on International conference on multimodal interaction*, 543–550.

Ko, B. C. (2018). A brief review of facial emotion recognition based on visual information. *Sensors*, 18(2), 401.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of IEEE*, 86(11), 2278-2324.

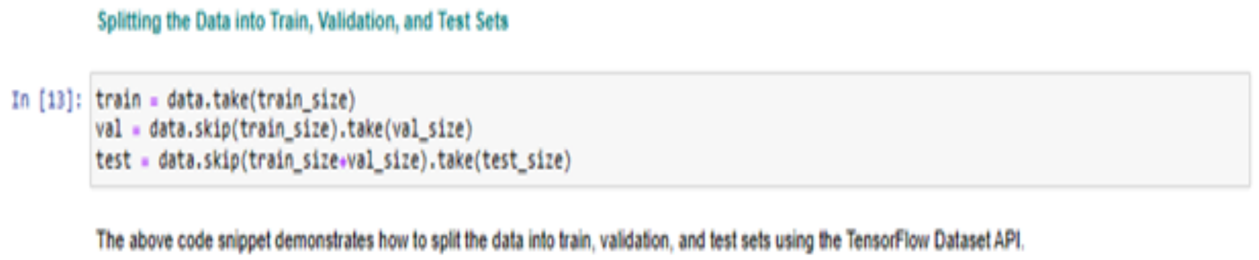
Li, S., & Deng, W. (2020). Deep facial expression recognition: A survey. *IEEE Transactions on Affective Computing*.

Lopes, A. T., de Aguiar, E., De Souza, A. F., & Oliveira-Santos, T. (2017). Facial expression recognition with convolutional neural networks: Coping with few data and training sample order. *Pattern Recognition*, 61, 610-628.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems* (pp. 8026-8037).

Zeng, Z., Pantic, M., Roisman, G. I., & Huang, T. S. (2009). A survey of affect recognition methods: Audio, visual and spontaneous expressions. *IEEE transactions on pattern analysis and machine intelligence*, 31(1), 39-58.

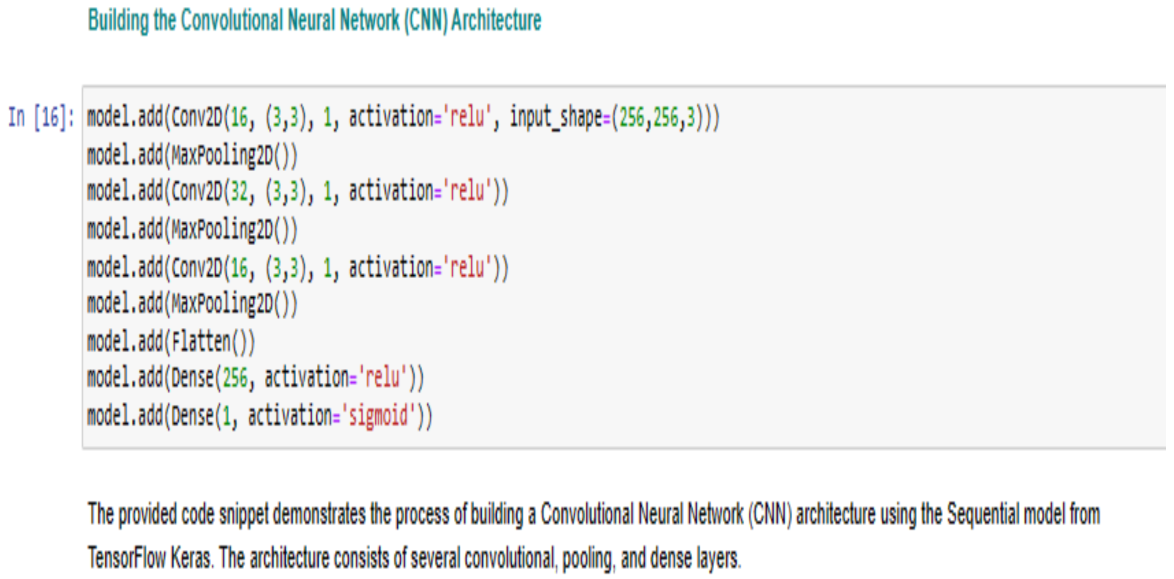
10. Appendices



Splitting the Data into Train, Validation, and Test Sets

```
In [13]: train = data.take(train_size)
         val = data.skip(train_size).take(val_size)
         test = data.skip(train_size+val_size).take(test_size)
```

The above code snippet demonstrates how to split the data into train, validation, and test sets using the TensorFlow Dataset API.



Building the Convolutional Neural Network (CNN) Architecture

```
In [16]: model.add(Conv2D(16, (3,3), 1, activation='relu', input_shape=(256,256,3)))
         model.add(MaxPooling2D())
         model.add(Conv2D(32, (3,3), 1, activation='relu'))
         model.add(MaxPooling2D())
         model.add(Conv2D(16, (3,3), 1, activation='relu'))
         model.add(MaxPooling2D())
         model.add(Flatten())
         model.add(Dense(256, activation='relu'))
         model.add(Dense(1, activation='sigmoid'))
```

The provided code snippet demonstrates the process of building a Convolutional Neural Network (CNN) architecture using the Sequential model from TensorFlow Keras. The architecture consists of several convolutional, pooling, and dense layers.