Leveraging Machine Learning for Defective Industrial Equipment Classification

Narendra Singh Dangi

Abstract— This study addresses the development of a machine learning model aimed at classifying images of industrial equipment into two categories: 'defective' and 'non-defective'. The objective involves selecting or creating a dataset comprising labeled images of industrial equipment, with additional annotations for defect types in the defective images. The proposed model leverages convolutional neural networks (CNNs), a potent deep learning architecture adept at extracting features directly from image pixels. Employing Python3 and utilizing the online available dataset as a reference, the study outlines the process of data preprocessing, model training, and evaluation. Key techniques such as 2D convolutional layers and Max-Pooling are utilized to extract pertinent features from the images, enabling robust training of the classification model. The model's performance is assessed using standard classification metrics like accuracy, precision, and recall. Results indicate a promising accuracy rate of 90.47% for the provided dataset.

***Keywords*- Convolution filters, Convolution layer, CNN, Image recognition, MAX pooling.**

I. Introduction

In industrial settings, the identification and classification of defective equipment play a pivotal role in ensuring operational efficiency, product quality, and workplace safety. Detecting defects manually can be time-consuming, prone to errors, and often requires expert knowledge. However, with advancements in machine learning and computer vision, automated systems can assist in this process by accurately classifying industrial equipment as either defective or non-defective based on image data.

The primary objective of this research is to develop a robust machine learning model capable of classifying images of industrial equipment into two main categories: 'defective' and 'non-defective'. This model aims to streamline the inspection process, enabling prompt identification of faulty equipment and facilitating timely maintenance or replacement, consequently reducing downtime and minimizing production losses.

Related Work

The Task of classification started with:

1. **Initial Approach**: The inital classification system starts with four fast-food classes. It initially segments images to form feature vectors that include size, shape, texture, color (normalized RGB), and other context-based features. Texture information is extracted using Gabor filter responses. However, this approach performs well for food replicas but less efficiently for real images due to variations in image size and capturing conditions.

2. **Scale Invariant Feature Transform (SIFT)**: To address the limitations of the initial approach, SIFT features have been extracted and experimented with on homemade foods, fast-food, and fruits. SIFT features offer better performance with fewer classes, despite having more images per class.

3. **Bag of Features (BoF)**: Inspired by the Bag of Words (BoW) approach in natural language processing, BoF is used in image analysis to capture common visual patterns in images. It reduces complexity and has been applied to food image recognition.

4. **Databases**: Having a high-quality database is crucial for CNN classification. Researchers have created real-time databases of images, serving as benchmarks for evaluating classification algorithms.

5. **Expanding Datasets**: Researchers have focused on collecting diverse datasets. One

dataset contains over 3,500 instances of Defective and non defactive. Local and global features are extracted and tested with various classifiers. Another dataset was created for mobile-based log systems, achieving an accuracy of 62%. Three-dimensional properties of image shapes have also been used for feature extraction.

6. **Deep Learning**: Deep Convolutional Neural Networks (CNNs) have gained popularity in food recognition. They have been trained and fine-tuned using datasets such as UEC-100, UEC-256, ImageNet, and ILSVRC. Some approaches use Global Average Pooling layers to generate Food Activation Maps (FAMs), which provide heat maps of food probability. Fine-tuning and thresholding are applied to generate bounding boxes for images items.

In summary, the field of classification has evolved from traditional feature-based methods to deep learning approaches, and researchers have focused on creating diverse datasets and improving classification accuracy, especially in real-world scenarios with varying conditions.

II. PROPOSED METHODOLOGY

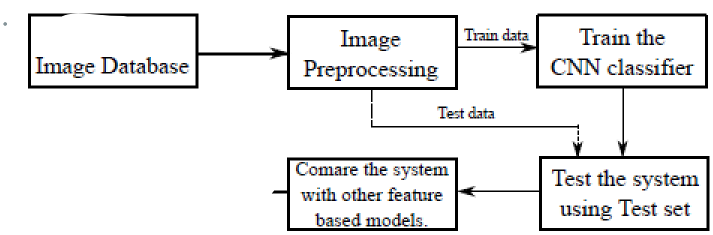

Fig. 1. The framework we have used to recognize the defactive and non defactive

The Dataset

Firstly, we have collected the data from Internet using it is easily availbale

1. **Dataset Purpose**: The reason for considering this dataset is to make the system more realistic. using this dataset we train our computer Vision machine learning model.
2. **Dataset Contents**: Each category in the dataset contains 3000 training data and 200 test data. This dataset have a set of 2 categories, and for each category, we have both training and testing data. The training data is typically used to teach a machine learning model, while the testing data is used to evaluate the model's performance.
3. **Labeling**: Care has been taken to label the training and testing images properly. Proper labeling is crucial in supervised machine learning tasks as it ensures that the model learns to associate the correct categories with the images.
4. **Image Rescaling**: The images in the dataset have been resized to a uniform size of 256x256 pixels. Image preprocessing, including resizing, is often performed to ensure that all images have the same dimensions, making them compatible with machine learning algorithms.

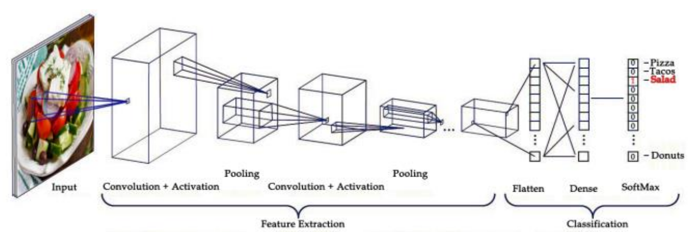

Fig. 2. A Simple CNN Architecture

Neural Network Configuration

1. **AvgPool**: The AveragePooling2D function, with a pool size of (2,2), serves two main purposes. First, it helps decrease the variability in the data, making it more stable for further processing. Second, it simplifies the computational workload by reducing the amount of information passed to the next layer. This layer then forwards its modified output to the subsequent layer in the neural network.
2. **Convolution:** an input with a size of (256,256,3), a Convolution2D function is employed. This layer's primary role is to

generate feature maps through a process called convolution, where it scans and combines the input data to create these maps.

3. **MaxPool:** A MaxPooling2D function is applied, which plays a crucial role in data processing. This pooling operation serves two key purposes. First, it helps diminish data variance, leading to a more stable representation. Second, it reduces computational complexity by down sampling the data. Max pooling specifically focuses on extracting critical features like edges and high-intensity regions, whereas average pooling tends to capture features more smoothly across the input.

4. **Dropout:** Dropout is a regularization method used to mitigate overfitting in neural networks by discouraging intricate co-dependencies among the training data. It provides an efficient means of conducting model averaging within neural networks. The concept of "dropout" involves randomly deactivating units, both in the hidden and visible layers of a neural network. In this context, a dropout scale of 0.3 has been specified, indicating that approximately 30% of the units are deactivated during each training iteration to enhance the model's generalization capability.

5. **Fully connected:** Fully connected layers, also known as dense layers, establish connections between every neuron in one layer to every neuron in another layer within a neural network.

6. **Softmax:** The Softmax function, when used as an output function, shares similarities with the Max layer, while also being suitable for gradient-based training. It achieves this by applying the exponential function to each value in the previous layer, enhancing the probability of the maximum value compared to the others. Additionally, the outputs of the Softmax function always sum up to 1.0, ensuring a valid probability distribution.

Image Processing to CNN

Several image preprocessing techniques are employed to optimize the proposed system, ensuring its effectiveness in classifying images from various angles. All images are standardized to a size of 256x256x3 pixels. The global average fusion function calculates the average of all image features. The dense() function determines the output state based on these averages. To mitigate overfitting issues, an input dropout rate of 0.3 is applied. Additionally, the softmax activation function is used to identify the real class from a set of multiple classes. It assigns the class based on the highest probability value and disregards the probabilities associated with other classes.

Neural Network Training

The basic CNN architecture employed in the proposed research is illustrated in Figure 2. To improve performance, Stochastic Gradient Descent (SGD) with a rapidly decreasing learning schedule is utilized. The model undergoes training for a total of 10 epochs, and three callbacks are configured to monitor and log its progress. Specifically, a learning rate scheduler is implemented, which takes the epoch index as input and adjusts the learning rate accordingly. Model checkpoints are created using the check pointer callback and saved as .Tflite files. Only the models with the best performance scores are retained and saved.

Usage of Neural Networks and Web Scraping

This subsection describes the use of neural networks and web scraping for the task of food classification.

1. **Image Augmentation:** One-hot encoding is used in this step to extract a set of binary features from each label. A feature that can accept any value from n- classes is preferable to this. A multiprocessing tool enables the GPU to be used to its full potential.

III. EVALUATION

In this section we have discussed the model building and model training techniques for food classification.

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 254, 254, 32)      896

 max_pooling2d (MaxPooling2D  (None, 127, 127, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 125, 125, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 62, 62, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 60, 60, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 30, 30, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 28, 28, 64)        73792

 max_pooling2d_3 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 flatten (Flatten)           (None, 12544)             0

 dense (Dense)               (None, 64)                802880

 dropout (Dropout)           (None, 64)                0

 dense_1 (Dense)             (None, 128)               8320

 dense_2 (Dense)             (None, 15)                1935

=================================================================
Total params: 980,175
Trainable params: 980,175
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3 Summery of the Neural Network.

```
history=model.fit(
    train_ds,
    batch_size=32,
    epochs=50,
)
```

```
Epoch 1/2
35/35 [==============================] - 226s 6s/step - loss: 5.7881 - accuracy: 0.0188
Epoch 2/2
35/35 [==============================] - 214s 6s/step - loss: 3.3990 - accuracy: 0.0464
```

Fig. 4. Model Under Training

Training the Model

1. We will train our model using 3000 images, each characterized by feature vectors with dimensions of 256x256x3.
2. We will determine the number of epochs, representing the iterations of the training dataset, with the goal of finding a balance where our model neither underfits nor overfits the data
3. We will save multiple models, with the final one reserved for future testing purposes.

IV. RESULT / ANALYSIS

In this section we have discussed the results, and the observations we have found while testing the model and the performance measurement techniques.

A. Evaluation of Models

To evaluate the model we have passed the our test data and Each class label will be plotted on a confusion matrix along with the percentage of times it was successfully categorized and the percentage of times it was wrongly labeled as a different class.
By the use of confusion matrix we have find the accuracy of model which comes 90.73%.

```
model.evaluate(test_ds)
```

```
7/7 [==============================] - 21s 86ms/step - loss: 0.1999 - accuracy: 0.9474
[0.19985297322273254, 0.9473684430122375]
```

Fig. 5. Test result

B. Output

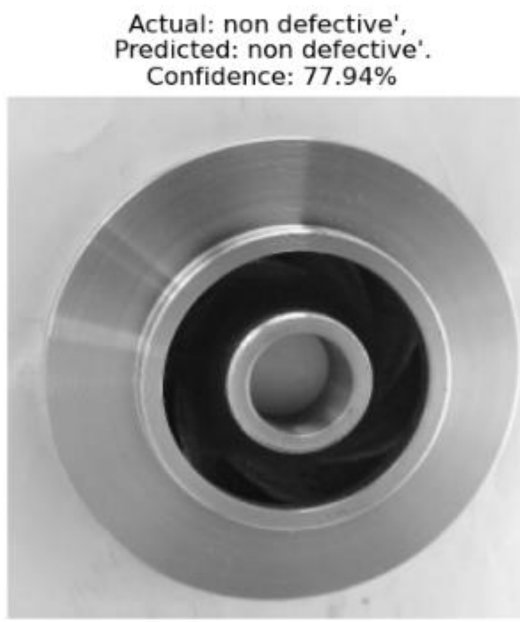

Fig. 6. image prediction using deep neural network

After predicting the food we have match the class of food to the nutrition value and the recipe in the nutrition dataset.

V. CONCLUSION AND FUTURE WORK

The system works really well and is okay to use. But, to make it work with big data sets of pictures and videos, we need really powerful computers. The system can learn from really complex data, but it takes a long time to teach it. However, once it's learned, it can give you results quickly. We make sure the pictures are prepared properly, and we test all sorts of pictures using this system. We found out that this system is best at sorting pictures when there are lots of different categories.

AUTHORS' PROFILES

Narendra Singh Dangi: 20bds036@iiitdwd.ac.in
Graduate in Data Science and Artificial Intelligence (2024) from Indian Institute of Information Technology, Dharwad Karnataka. certified Machine Learning Enthusiast by Microsoft and has done many ML projects based on Image and Text classification using Deep Neural Network(CNN, RNN, and LSTM), Some of the projects are Image Caption Generator, Image classification, and Sentiment Analysis.

REFERENCES

[1] W. Wu and J. Yang, "Fast food recognition from videos of eating for calorie estimation," in Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on. IEEE, 2009, pp. 1210–1213.

[2] N. Yao, R. J. Sclabassi, Q. Liu, J. Yang, J. D. Fernstrom, M. H. Fernstrom, and M. Sun, "A video processing approach to the study of obesity," in Multimedia and Expo, 2007 IEEE International Conference on. IEEE, 2007, pp. 1727–1730.

[3] S. Yang, M. Chen, D. Pomerleau, and R. Sukthankar, "Food recognition using statistics of pairwise local features," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010, pp. 2249–2256.

[4] M. Bosch, F. Zhu, N. Khanna, C. J. Boushey, and E. J. Delp, "Combining global and local features for food identification in dietary assessment," in Image Processing (ICIP), 2011 18th IEEE International Conference on. IEEE, 2011, pp. 1789–1792.

[5] M. M. Anthimopoulos, L. Gianola, L. Scarnato, P. Diem, and S. G. Mougiakakou, "A food recognition system for diabetic patients based on an optimized bag-of-features model," IEEE journal of biomedical and health informatics, vol. 18, no. 4, pp. 1261–1271, 2014.

[6] G. Shroff, A. Smailagic, and D. P. Siewiorek, "Wearable context-aware food recognition for calorie monitoring," in Wearable Computers, 2008. ISWC 2008. 12th IEEE International Symposium on. IEEE, 2008, pp. 119–120.

[7] F. Zhu, M. Bosch, I. Woo, S. Kim, C. J. Boushey, D. S. Ebert, and E. J. Delp, "The use of mobile devices in aiding dietary assessment and evaluation," IEEE journal of selected topics in signal processing, vol. 4, no. 4, pp. 756–766, 2010.

[8] F. Kong and J. Tan, "Dietcam: Automatic dietary assessment with mobile camera phones," Pervasive and Mobile Computing, vol. 8, no. 1, pp. 147–163, 2012.

[9] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 2. IEEE, 2005, pp. 524–531.

[10] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," Machine learning: ECML-98, pp. 137–142, 1998.

[11] M. Puri, Z. Zhu, Q. Yu, A. Divakaran, and H. Sawhney, "Recognition and volume estimation of food intake using a mobile device," in Applications of Computer Vision (WACV), 2009 Workshop on. IEEE, 2009, pp. 1–8.

[12] M. Chen, K. Dhingra, W. Wu, L. Yang, R. Sukthankar, and J. Yang, "Pfid: Pittsburgh fast-food image dataset," in Image Processing (ICIP), 2009 16th IEEE International Conference on. IEEE, 2009, pp. 289–292.