# Food Detection by Fine-Tuning Pre-trained Convolutional Neural Network Using Noisy Labels

**Shroog Alshomrani, Lina Aljoudi, Banan Aljabri, Sarah Al-Shareef,**
*{s44280319, s44280160, s44280163}@st.uqu.edu.sa, saashareef@uqu.edu.sa*
Computer Science Department, Umm Al-Qura University, Makkah, Saudi Arabia

**Summary**
Deep learning is an advanced technology for large-scale data analysis, with numerous promising cases like image processing, object detection and significantly more. It becomes customarily to use transfer learning and fine-tune a pre-trained CNN model for most image recognition tasks. Having people taking photos and tag themselves provides a valuable resource of in-data. However, these tags and labels might be noisy as people who annotate these images might not be experts. This paper aims to explore the impact of noisy labels on fine-tuning pre-trained CNN models. Such effect is measured on a food recognition task using Food101 as a benchmark. Four pre-trained CNN models are included in this study: InceptionV3, VGG19, MobileNetV2 and DenseNet121. Symmetric label noise will be added with different ratios. In all cases, models based on DenseNet121 outperformed the other models. When noisy labels were introduced to the data, the performance of all models degraded almost linearly with the amount of added noise.

***Keywords:***
*deep learning, food image detection, symmetric label noise, convolutional neural networks, transfer learning.*

## 1. Introduction

Healthy diets are essential for human health. In the current time, people have increased awareness of the importance of a healthy diet. As a result, automatic recognition systems for drinks and foods have appeared. These systems recognise the components of the drink or food and estimate their nutritional value to help evaluate the diet. The group benefiting from these systems is significant, including patients with dietary restrictions and people who follow a diet [1].

Several factors make detecting images of food and drink difficult. Among them, foods are usually deformable objects, and thus it is difficult to determine the type of food in the image. Besides, some food items may have a high intra-class variance but a low inter-class variance [2]. In other words, objects of the same food category look very different but resemble other items from different categories. Consequently, detecting a food category for such items become challenging.

Previously, machine learning was active in many areas and worked effectively for data processing; however, it cannot analyse raw data because it usually needs to be supplemented with a manual feature extraction method. Through advancements in hardware computing capacity and storage space, machine learning capabilities can be enhanced by adding more complex structures to represent unstructured data with deep models. Due to the strong learning ability of the deep learning method in regression and classification, solving many complex problems become quick and effective. However, it requires a sufficient amount of data for the required problem. For example, due to the powerful feature of automatic learning features, deep learning is applied in the food domain to classify a food category to discover its quality and estimate the number of food calories. Convolutional Neural Networks (CNNs) are currently considered one of the most common deep models used in analysing big data in various fields of computer vision research [3].

CNNs are the type of deep neural networks inspired by the visual cortex of animals as these individual neurons interact with the overlapping regions of the visual field. Consequently, CNN becomes suitable for computer vision because the goal of computer vision is similar to animal vision: gaining an understanding of interfering images [4]. Furthermore, we implemented CNN in this paper because it can ignore surrounding noise for label noise with enough training data, because of its high classification accuracy and learn optimal features from images adaptively; thus, this suitable for our research type. Moreover, CNN can be trained by the way that it can detect objects in the same network [5, 6].

This research implemented four fine-tuned pre-trained CNN models on the Food-101 data set with different ratios of added label noise. Also, it experimented with different architectures using two types of optimisers at different learning rates.

The rest of this paper is ordered as follows. Section 2 reviews the related literature on the subject and similar previous attempts. Then, the Food-101 data set is described in Section 4. An overview of the pre-trained models used in this study and how they were fine-tuned is presented in Section 3. Section 5 describes this study's experimental design and configurations with the results its discussion in Section 6. Finally, this study is concluded in Section 7 with a brief direction for future work.

Manuscript received July 5, 2021
Manuscript revised July 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.7.22

**Table 1**
Performance of previous work using deep models on Food-101 data set. Top-1 and Top-5 are accuracy scores for the model when considering the first predicted class or top five classes; respectively.

| Reference | #Class | Additional data | Data Augment. | Pretrained models | Top-1 | Top-5 |
|---|---|---|---|---|---|---|
| Heravi et al. [10] | 101 | UECFood-256 | Yes | ConvNet, AlexNet, GoogLeNet, VGG, ResNet | 65.40% | 87.00% |
| Yanai and Kawano [19] | 100 | Twitter photo data | Yes | - | 70.40% | -- |
| Wu et al. [11] | 101 | 5-Chain | No | GoogLeNet | 72.10% | -- |
| Pandey et al. [12] | 101 | IndianFood | No | GoogleNet, AlexNet, ResNet | 72.10% | 91.60% |
| Liu et al [13] | 101 | -- | No | GoogLeNet | 77.40% | 93.70% |
| Liu et al [14] | 101 | UEC dataset | No | Inception | 77.00% | 94.00% |
| Fu et al. [20] | 100/256/101 | UEC100, UEC256 | Yes | - | 78.50% | 94.10% |
| Ciocca et al. [21] | 1200 | UNICT-FD1200 | No | ResNet-50 | 82.50% | 95.80% |
| Zheng et al. [16] | 101 | UECFood-256 | No | AlexNet, InceptionV3 | 88.00% | -- |
| Hassannejad et al. [17] | 101 | UECFOOD100, UECFOOD256 | No | InceptionV3 | 88.30% | 96.90% |
| Martinel et al. [18] | 101 | UECFOOD100, UECFOOD256 | Yes | - | 90.30% | 98.70% |

## 2. Related Work

The reviewed literature can be grouped into two collections: food detection using deep models and training models using noisy labels.

### 2.1 Food Detection using Deep Models

In 2019, Zhou et al. [7] reviewed most of the problems addressed by machine learning researchers in the food domain. Their survey found that deep learning models were more effective than other methods like traditional machine learning algorithms and manual feature extractors.

In 2014, Kagaya et al. [8] applied CNN model to distinguish food images from non-food ones. They used two data sets: ImageNet [9] and a collection of food images collected from the food-logging app. They experimented with different depth of CNN, between 2 to 4, and found that a two-layer CNN architecture outperformed the rest with an accuracy of 93.8%. They also found that food colours dominate food recognition.

In 2018, Heravi et al. [10] proposed the ConvNet system to reduce the number of parameters used in training while maintaining high accuracy and was tested on two real-world datasets UECFood-256 and Food-101. This system gave high results compared to ConvNet, AlexNet, GoogLeNet, VGGNet, but it is close to the efficiency of ResNet50.

In 2016, Wu et al. [11] proposed an error improvement system to be closer to correct by adding hierarchical semantic relationships to CNN. The proposed method has been tested with GoogLeNet model on two datasets: Food-101 and 5-Chain menu from popular restaurants. This system improved the results and reduced loss function.

In 2017, Pandey et al.[12] proposed a system to identify the meal's contents called Ensemble Net, which combines the outputs of three Pre-trained models GoogLeNet and AlexNet and ResNet. They applied this system to two sets of data from the real world. They found that it outperforms many other methodologies.

In 2016, Liu et al. [13] proposed a system to enhance the accuracy of dietary assessment by analysing the food images captured by mobile devices. They proposed CNN-based algorithms with optimisations, and they applied them to two real-world food image datasets. Their results proved that the proposed approach is a promising solution for the food image classification problem.

In 2017, Liu et al.[14] proposed a system for identifying food and helping with dietary assessment by capture photos by mobile, cleaning them, pre-processing them, and then analysing them using CNN models-Inception. They applied this system to two data sets, Food-101 and a UEC data set. They found that this system gave high results compared to their other proposed system in [13].

In 2020, Ramdani et al. [15] proposed a food detection system by using CNN to help to automate the estimation of food price. They used 480 for fine-tuning with 80% of the data for fine-tuning and 20% for testing. Their system can classify six types of food and achieve an accuracy of 100% for the six types and with 10 seconds of detection time.

In 2018 Zheng et al.[16] proposed a new framework depending on two approaches: the mid-level and deep CNN approach for food image recognition using three datasets. The researchers faced difficulty while training the model of CNN with the unlabeled mid-level parts data. They solved this problem by designing a clustering-based FP label mining scheme that used unlabeled data to generate part-level labels. The proposed approach achieved excellent accuracy when comparing it with other methods.

In 2016 Hassannejad et al. [17] introduced a pre-trained model of deep CNN called Inception V3. They used three kinds of dataset: ETH Food-101, UEC FOOD 100, and UEC FOOD 256. They achieved 88.28%, 81.45%, and 76.17% respectively for top-1 accuracy and 96.88%, 97.27%, and 92.58% for the top-5 accuracy.

Another research for food recognition in 2018 was proposed by Martinel et al. [18] depending on deep neural network (DNN) with two primary branches: the residual and slice networks. They used three datasets which were: UECFood100, UECFood256 and Food-101. This wide slice residual network (WISer) architecture achieved a good performance compared to other existing architectures.

In 2015, Yanai and Y. Kawano.[19] studied the efficacy of using the deep CNN to detect food images on two datasets: UEC-FOOD100 and UEC-FOOD256. They use different techniques like pre-training with ImageNet data, fine-tuning and extract features from the pre-trained CNN. The result has shown the effectiveness of DCNN for large-scale image data by 78.77% and 67.57% accuracy for the UEC-FOOD100/256 datasets.

In 2017, Z. Fu, D. Chen, and H. Li.[20] introduced the baseline approach, it is a robust deep network for food images. The method implemented on ChinFood1000 along evaluated on three famous datasets: UEC100, UEC256, Food-101. All results prove the effectiveness of their approach.

As this work uses the Food-101 data set, Table 1 summarises all the previous studies that use this data set for food detection and classification tasks.

### 2.2 Noisy Label

Several studies have been investigating overcoming the noisy label issue when training deep models. These studies have attempted to make the models robust to noise in labels or de-noise the data set before the training. Of the former set of studies, Li et al. [22] proposed an approach called Cyclic Annealing Training (CAT) which can speed up the CNN training in every M-step by utilising a fast annealing training method. Thus, it reduces the training time and improves the performance of the image classification. CAT uses three kinds of datasets: MNIST, CIFAR-10 and CIFAR-applies under different noisy labels pattern; 46% of noisy labels on the MNIST dataset randomly flip the labels. The random flipping follows the pattern [7,9,0,4,2,1,3,5,6,8], which means digital 0 will be labelled by 7, 1 by 9, and so on. CAT achieved a classification accuracy of 99.77%. CIFAR-10 dataset trained on 10% randomly flipped labels, and CIFAR-100 with 50% flipped label noise. When comparing the CAT approach with an expectation-maximisation (EM), the results showed that the CAT approach needs less time to converge, increasing the CNN effectiveness and making it more robust. At the same time, EM requires too much time costs.

On the other hand, Arazo et al. [23] suggested a training approach for CNNs which avoided fitting noisy labels. Their strategy relied on unsupervised learning to distinguish noisy and true labels. In other words, it cleaned the training data on the fly. They applied their approach on CIFAR-10 and CIFAR-100 data sets to display their approach's strengths and weaknesses and prove its outstanding performance.

Fewer studies were exploring the behaviour of deep models when they were trained using noisy labels. An example of such a study is the work of Rolnick et al. [24]. They used three data sets: MNIST, ImageNet and CIFAR-10 with different ratios of noisy labels. These noises were added using three structures: confusing order, reverse confusing order, and random order. Then, they applied CNN with Conv4, Conv6 and ResNet. They claimed that the performance of the CNN model with noisy labels depends on the amount of noise in the set, batch size and learning rate. Overall, ResNet outperformed the other CNN models.

In this study, the performance and behaviour of fine-tuning CNN models will be investigated when using in-domain data with noisy labels. Four pre-trained models will be understudied: InceptionV3, VGG19, MobileNetV2 and DenseNet121. Similar to the work of Rolnick et al. [24], the label noise will be added artificially to a clean data set with different ratios, as will be described in Section 3.

## 3. Methodology

Most models are trained on a clean data set; however, this might reduce generalisation and overfitting when the data set is small. One solution is to use a pre-trained model previously trained on an extensive data set for another task. Then, fine-tune this model using an in-domain data set. This work focuses on a multi-classification food detection task using pre-trained models fine-tuned by data set with noisy labels. First, the noise was added to the labels using the method described in Section 3.A. Next, the four pre-trained models, InceptionV3, MobileNetV2, VGG19 and DenseNet121, are described in the following section.

### 3.1 Synthetic Label Noise

Real-world data contains a lot of noise, whether in samples or their labels. As this work attempt to measure the impact of label noise in fine-tuning pre-trained models, synthetic noisy labels are added to the labels of a clean data set with controlling ratios. Generally, this can be done by randomly flipping the original labels. The flipped labels can be either:

- Symmetric label noise is a class independent noise, where noise ratio is the probability of a label flip spread uniformly among all the other classes.

- Asymmetric label noise is a class dependent noise, where a noise ratio is a probability if a label flips to a specific class.

In this study, only symmetric label noise was used. In this case, for a multi-classification problem with $C$ classes, assume that the $k^{\text{th}}$ sample in the data set has an actual label $\hat{y}_k = i$. When the noise ratio $\epsilon$ is applied to the data set, the new (noisy) label can be assigned randomly using the following probability:

$$p(y_k = j) = \frac{\epsilon}{C-1}, \forall j \neq i;$$

$$p(y_k = i) = i - \epsilon;$$

In other words, there is an equal chance to assign incorrect label between all non-true labels.

## 3.2 CNN Pre-trained Model Architectures

CNN is a method in deep learning to detect objects, shapes and edges by a sequence of filters, also known as kernels consisting of trainable parameters, which convolves the input images to extract the features. This research used four types of pre-trained models which are: InceptionV3 [25], MobileNetV2 [26], VGG19 [27] and DenseNet121 [28].

Inception [29] model has 22 layers, and the main property of this architecture is the optimised utilisation of the computing resources inside the net. Its design permits the depth and width increasing of the network while preserving the computational cost constant. InceptionV3, also known as GoogLeNet, is one of the Inceptions family with 48 layers deep and many improvements [25]. MobileNet [30] is based on depthwise separable convolutions and consists of 28 layers. Its architecture is small and less computationally expensive. This study uses MobileNetV2 [26], which still uses depthwise separable convolutions and has 53 layers. It permits a very memory-efficient inference and relies on utilised standard operations present in all neural frameworks. VGG [27] network's architecture aims to evaluate the increasing depth of networks using an architecture with small (3 × 3) convolution filters. There are multiple versions of the VGG that varies in depth and number of layer. VGG19 is the one used in this study with 19 layers. DenseNet [28] uses feed-forward to introduce direct connections between any two layers while using the same feature-map size. It consists of 5 layers, and it helps to decrease the vanishing-gradient problem and reduce the number of parameters. This study uses one version of DenseNet with 121 layers, also known as DenseNet121. Table 2 summarises the specifications of the used pre-trained models in this study. As shown, VGG19 has the most significant number of trainable parameters with the lowest number of layers. On the other hand, DenseNet121 is the deepest model but one with fewer trainable parameters.

**Table 2**
Specifications of the pretrained models used in this study. Params. indicates to the number of trainable parameters. Top-1 and Top-5 is the model performance on image classification task of ImageNet [9] Challenge.

| **Model** | **Layers** | **Params. (M)** | **Size (MB)** | **Top-1 %** | **Top-5 %** |
|---|---|---|---|---|---|
| InceptionV3 [25] | 48 | 24 | 92 | 79.0 | 94.5 |
| DesneNet121 [28] | 121 | 8 | 33 | 75.0 | 92.3 |
| MobileNetV2 [26] | 53 | 4 | 14 | 74.7 | -- |
| VGG19 [27] | 19 | 144 | 549 | 74.5 | 92.0 |

## 3.3 Fine-tuning Pre-trained Models

Fine-tuning is the process of using a model which trained for a specific task to another task, also known as a pre-trained model. Several strategies are to be followed in the literature depending on the size of the in-domain data set. If the in-domain data set is small, the output layer of the pre-trained model is removed. Then, the model is used to extract feature maps from the in-domain data. These feature maps are then fed to either a classifier or one or more fully connected layer(s). If the in-domain data set is large enough, the output layer is replaced with a suitable SoftMax layer along with one or more fully connected layers if needed. Then, the whole model is trained with or without freezing some of the original layers during the training. If the layer is not frozen, its pre-trained weights will be used as initial values for the training process. The latter strategy was used in this study as there is enough data for each class in the data set. The output layer was replaced with a global average pooling layer to extract the most significant features, two fully connected layers with 20% of dropout and 320 nodes each and an output layer with some nodes equivalent to the number of classes to be detected. Rectified linear activation function (ReLU) was used as an activation function for the first two added fully connected layers, and SoftMax was used for the last one. Figure 1 illustrated the architecture of the entire fine-tuned CNN model when MobileNetV2 was used as the pre-trained model.

## 4. Food-101 Data Set

Food-101 data set [31] contains 101 food categories. Each category has 1000 images, split into 750 images for training and 250 images for testing. Labels for the testing set have been manually cleaned, unlike labels for the training set, which contain some noise. Each image is a square image with a height and width of 512. Figure 2 shows a sample from the data set with their labels.

The main challenge in this data set is that images from the same food category might look very different, that is, high intra-class variance. In contrast, images from different food categories might look similar, i.e. low inter-class variance. Greek salad is an example of a category with low inter-class variance as it looks very different based on its origin and the used ingredients, as shown in Figure 3. On the other hand, steak and chocolate cake dishes look very similar, as shown in Figure 4.

All images were resized to be 224×224 and normalised to have pixel value to within [0,1] range for this study.

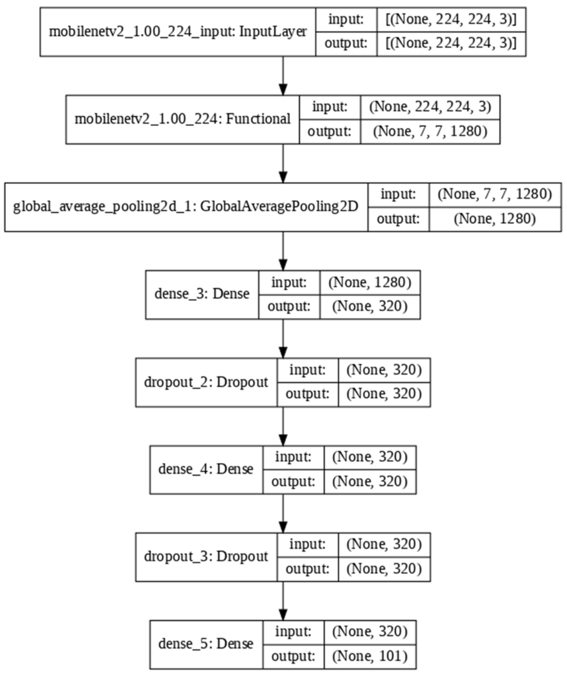

Figure 1: The architecture of the entire fine-tuned CNN model when MobileNetV2 was used as the pretrained model.

Figure 2: Four samples from Food-101 data set with their labels.

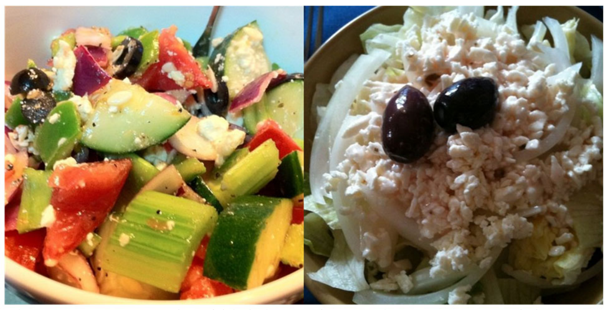

Figure 3: An example of high intra-class variance. Images of the same food category (greek salad) that look different.

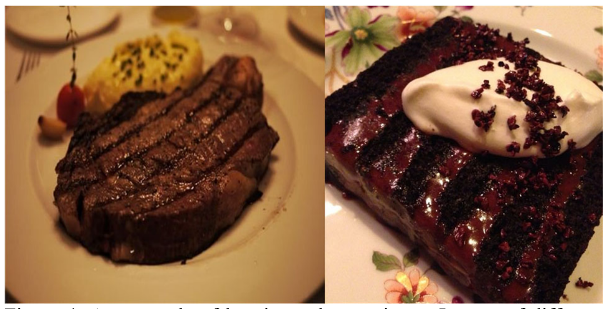

Figure 4: An example of low inter-class variance. Images of different food categories (left: steak, right: chocolate cake) that look similar.

Table 3
Details of data augmentation applied during training.

| **Argument** | **Value** |
|---|---|
| Horizontal flip | Yes |
| Rotation range | 40 |
| Width shift rnage | 20% |
| Height shift range | 20% |
| Zoom range | 20% |

## 5. Experiments

### 5.1 Experimental Design

All experiments were implemented and evaluated in Python and leverage TensorFlow and Keras [32] using Google Colab's Jupyter Notebooks [33] environment. First, data were split after shuffling into three sets: training, validation and testing with ratios 75%, 12.5% and 12.5%, respectively. Next, symmetric label noise was applied with different ratios, 10-30\%, using Chen et al. [34].

Data augmentation have been applied during training to increase the diversity of the data and improve the model's generalisation and minimise overfitting. It introduces random transformations at every epoch, such as rotation, horizontal flip, zoom, width/height shift and filling boundaries with the nearest colour. Table 3 lists the detailed parameters for data augmentation.

Table 4
Details of chosen training hyperparameters.

| Hyperparameter | Value |
|---|---|
| Batch size | 32 |
| Learning rate | 0.001 and 0.0001 |
| Momentum | 0.8 |
| Epochs | variable |
| Early stopping | 100 |
| Optimizer | SGD and Adam |
| Momentum SGD | 0.9 |
| Learning rate decay | epoch/iteration |

Each model was investigated in two scenarios: fine-tuned using clean labels and fine-tuned using noisy labels. These two scenarios were used to monitor the model's performance and the impact of added noise on the labels. To fine-tune or train these models, first, a batch size of 32 was used. Second, each experiment was performed using two learning rates, 0.001 and 0.0001, with two different optimisers, stochastic gradient descent (SGD) and adaptive moment estimation (Adam). Table 4 summarises the chosen values for all hyperparameters, which were chosen empirically.

## 5.2 Evaluation Matrices

The performance of a multi-class classifier can be measured using average accuracy per class. For a multi-class classifier with $C$ classes, the average accuracy per class, $Acc$, can be computed as:

$$Acc = \frac{\sum_{i=1}^{C} A_i}{C};$$

where $A_i$ is the accuracy for class $i$, and $1 < i < C$, can be computed as follows:

$$A_i = \frac{tp_i + tn_i}{tp_i + fp_i + tn_i + fn_i}$$

Where $tp_i$ and $tn_i$ are the number of samples correctly detected to belong to class $i$ or not, also known as true positives and true negatives, respectively. $fp_i$ and $fn_i$ are the number of samples incorrectly identified to be in class $i$ or not, also known as false positives and false negatives, respectively.

In addition to accuracy, other metrics can be used to evaluate a classifier performance, such as precision and recall. Precision is measured from the model point of view, while recall is measured from the actual data point of view. Precision is defined as how much of the samples predicted to be in class $i$ are correctly classified as $i$. Recall, also known as sensitivity, is how much of the actual class $i$ samples were correctly classified. Average precision, $Pre$, and recall, $Rec$, for a classifier with $C$ classes can be computed as:

$$Pre = \frac{1}{C}\sum_{i=1}^{C} P_i,$$
$$Rec = \frac{1}{C}\sum_{i=1}^{C} R_i,$$

where $P_i$ and $R_i$ are per class precision and recall; respectively, which can be computed as follows:

$$P_i = \frac{tp_i}{tp_i + fp_i},$$
$$R_i = \frac{tp_i}{tp_i + fn_i},$$

While precision and recall measure two different aspects of a classifier, the F1-score considers both aspects into a single metric. F1-score, $F1$, is computed as the harmonic mean of the precision and recall:

$$F1 = \frac{2 \times Pre \times Rec}{Pre + Rec}$$

## 6. Results and Discussion

First, several experiments were performed using different combinations of hyperparameter, shown in Table 4. The best performing experiments are shown in Table 5.

**Table 5**
Details and performance of best models when fine-tuning using data with clean labels.

| Model | epochs | Learning rate | Optimizer | Accuracy % | F1 % | Recall % | Precision % | Top-5 % |
|---|---|---|---|---|---|---|---|---|
| InceptionV3 | 10 | 0.001 | SGD | 74.1 | 75.8 | 69.3 | 83.9 | 93.5 |
| | 10 | 0.0001 | Adam | 73.4 | 74.9 | 68.7 | 82.7 | -- |
| | 26 | 0.001 | SGD | **79.2** | **80.5** | **76.4** | **85.3** | -- |
| VGG19 | 10 | 0.001 | SGD | 64.9 | 65.7 | 55.4 | 81.4 | 90.5 |
| | 20 | 0.001 | SGD | 74.8 | 76.1 | 69.8 | **83.9** | -- |
| | 26 | 0.001 | SGD | **76.3** | **77.8** | **72.7** | **83.9** | -- |
| MobileNetV2 | 10 | 0.001 | SGD | 71.5 | 73.0 | 66.3 | 81.5 | -- |
| | 26 | 0.001 | SGD | **78.3** | **79.7** | **75.1** | **85.0** | 93.5 |
| | 10 | 0.001 | Adam | 66.4 | 67.4 | 57.0 | 83.2 | -- |
| DenseNet121 | 22 | 0.0001 | Adam | **81.7** | **82.7** | **79.3** | 86.4 | 94.6 |
| | 30 | 0.0001 | SGD | 75.4 | 76.4 | 68.7 | **86.5** | -- |

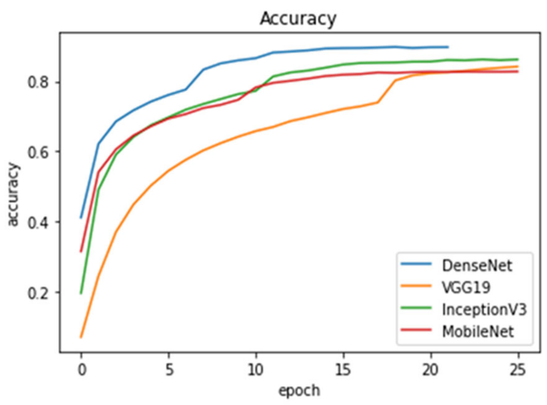

**Figure 5:** Learning curves for the accuracy of best performing fine-tuned models during training.

Generally, results are getting better for all models with more extended training, more epochs, and SGD optimiser except for DenseNet121. Precision ranges, for all experiments, between 81-87%, even when the accuracy is as low as 64.9%. Unlike precision, the recall has a broader range between 55-79%. These ranges might suggest that the models captured most of the classes, but they still confused them. The model based on DenseNet121 outperformed all other models with an accuracy of 81.7%. This performance could be due to its architecture which has a strong gradient flow for error signal between earlier layers and classification layer. Also, DenesNet121 received all previous layers as input, making the features more diverse and getting more rich patterns. In addition, it uses features of all levels of complexity, which leads to smoother decision boundaries that help to learn to discriminate between classes more effectively than the other models, as shown in Figure 5.

The configurations of these best models were used for the second set of experiments with noisy labels. Table 6 shows the evaluation of the models when fine-tuned with data, including different ratios of label noise. Consequently, there is a clear linear relationship with a negative correlation between all evaluation metrics and the amount of added symmetric label noise to the data, as depicted in Figure 6. In other words, the higher the noise ratio, the worse the model will perform.

## 7. Conclusion and Future Work

This paper aimed to explore the impact of label noise on fine-tuning pre-trained CCN models. The impact was measured in a food recognition task using Food-101 as a benchmark. Four pre-trained CNN models were included in this study: InceptionV3, VGG19, MobileNetV2 and DenseNet121. Symmetric label noise was added in three ratios: 10%, 20% and 30%. In clean data, the models based on DenseNet121 outperformed the other models. However, when noisy labels were introduced, the performance of all models degraded almost linearly with the amount of added noise. As a next step, methods for overcoming such noisy labels will be explored.

**Table 6**
Performance of best models when fine-tuning using data with different ratios of label noise.

| Model | Noise % | Accuracy % | F1 % | Recall % | Precision % |
|---|---|---|---|---|---|
| InceptionV3 | 0 | 79.2 | 80.5 | 76.4 | 85.3 |
| | 10 | 70.1 | 70.6 | 63.2 | 80.2 |
| | 20 | 62.6 | 61.6 | 53.1 | 73.9 |
| | 30 | 54.2 | 51.2 | 42.2 | 65.7 |
| VGG19 | 0 | 76.3 | 77.8 | 72.7 | 83.9 |
| | 10 | 67.9 | 66.6 | 56.7 | 81.2 |
| | 20 | 58.8 | 53.1 | 41.6 | 74.6 |
| | 30 | 51.2 | 41.1 | 29.9 | 67.2 |
| MobileNetV2 | 0 | 78.3 | 79.7 | 75.1 | 85.0 |
| | 10 | 70.1 | 69.6 | 61.6 | 80.6 |
| | 20 | 59.9 | 57.5 | 47.3 | 74.1 |
| | 30 | 51.4 | 45.7 | 35.3 | 65.9 |
| DenseNet121 | 0 | 81.7 | 82.7 | 79.3 | 86.4 |
| | 10 | 73.3 | 73.6 | 67.3 | 81.5 |
| | 20 | 64.5 | 63.8 | 56.7 | 73.2 |
| | 30 | 56.5 | 53.5 | 44.8 | 67.0 |

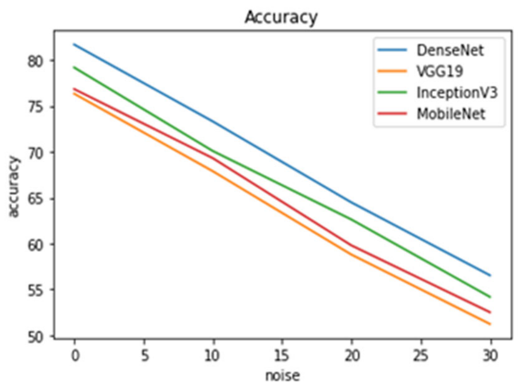

**Figure 6:** Accuracy of fine-tuned models in relation to the amount of added label symmetric noise.

## References

[1] D. de Ridder, F. Kroese, C. Evers, M. Adriaanse, and M. Gillebaart, “Healthy diet: Health impact, prevalence, correlates, and interventions,” Psychology & health,vol. 32, no. 8, pp. 907–941, 2017.
[2] S. Mezgec and B. Korouˇsi´c Seljak, "Nutrinet: a deep learning food and drink image recognition system for dietary assessment," Nutrients, vol. 9, no. 7, p. 657, 201
[3] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural networks, vol. 61, pp. 85–117, 2015.
[4] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," The Journal of physiology, vol. 160, no. 1, pp. 106–154, 1962.
[5] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localisation and

detection using convolutional networks," arXiv preprint arXiv:1312.6229, 2013.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed,C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in European conference on computer vision.Springer, 2016, pp. 21–37.

[7] L. Zhou, C. Zhang, F. Liu, Z. Qiu, and Y. He, "Application of deep learning in food: a review," Comprehensive reviews in food science and food safety, vol. 18, no. 6, pp. 1793–1811, 2019.

[8] H. Kagaya, K. Aizawa, and M. Ogawa, "Food detection and recognition using convolutional neural network," in Proceedings of the 22nd ACM international conference on Multimedia, 2014, pp. 1085–1088.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009, pp. 248–255.

[10] E. J. Heravi, H. H. Aghdam, and D. Puig, "An optimised convolutional neural network with bottleneck and spatial pyramid pooling layers for classification of foods," Pattern Recognition Letters, vol. 105, pp. 50–58, 2018.

[11] H. Wu, M. Merler, R. Uceda-Sosa, and J. R. Smith, "Learning to make better mistakes: Semantics-aware visual food recognition," in Proceedings of the 24th ACM international conference on Multimedia, 2016, pp. 172–176.

[12] P. Pandey, A. Deepthi, B. Mandal, and N. B. Puhan, "Foodnet: Recognising foods using an ensemble of deep networks," IEEE Signal Processing Letters, vol. 24, no. 12, pp. 1758–1762, 2017.

[13] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, andY. Ma, "Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment," in International Conference on Smart Homes and HealthTelematics. Springer, 2016, pp. 37–48.

[14] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, M. Yun-sheng, S. Chen, and P. Hou, "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," IEEE Transactions on Services Computing, vol. 11, no. 2, pp. 249–261,2017.

[15] A. Ramdani, A. Virgono, and C. Setianingsih, "Fooddetection with image processing using convolutional neural network (CNN) method," in 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT). IEEE, 2020, pp.91–96.

[16] J. Zheng, L. Zou, and Z. J. Wang, "Mid-level deep food part mining for food image recognition," IET ComputerVision, vol. 12, no. 3, pp. 298–304, 2018.

[17] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Mu-nari, M. Mordonini, and S. Cagnoni, "Food image recognition using very deep convolutional networks," in Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management, 2016, pp. 41–49.

[18] N. Martinel, G. L. Foresti, and C. Micheloni, "Wide-slice residual networks for food recognition," in 2018 IEEEWinter Conference on applications of computer vision (WACV). IEEE, 2018, pp. 567–576.

[19] K. Yanai and Y. Kawano, "Food image recognition using deep convolutional network with pre-training and fine-tuning," in2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). IEEE, 2015, pp. 1–6.

[20] Z. Fu, D. Chen, and H. Li, "Chinfood1000: A large benchmark dataset for Chinese food recognition," International Conference on Intelligent Computing. Springer, 2017, pp. 273–281.

[21] G. Ciocca, P. Napoletano, and R. Schettini, "Cnn-based features for retrieval and classification of food images," Computer Vision and Image Understanding, vol. 176, pp.70–77, 2018.

[22] J. Li, T. Dai, Q. Tang, Y. Xing, and S.-T. Xia, "Cyclic annealing training convolutional neural networks for image classification with noisy labels," in2018 25th IEEE International Conference on Image Processing (ICIP).IEEE, 2018, pp. 21–25.

[23] E. Arazo, D. Ortego, P. Albert, N. O'Connor, and K. McGuinness, "Unsupervised label noise modelling and loss correction," international Conference on MachineLearning. PMLR, 2019, pp. 312–321.

[24] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, "Deep learning is robust to massive label noise," arXiv preprintarXiv:1705.10694, 2017.

[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed,D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[26] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," CoRR, vol. abs/1801.04381,2018.[Online]. Available: http://arxiv.org/abs/1801.04381

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXivpreprint arXiv:1409.1556, 2014.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[29] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, andZ. Wojna, "Rethinking the inception architecture for computer vision," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp.2818–2826.

[30] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko,W. Wang, T. Wey and, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXivpreprintarXiv:1704.04861, 2017.

[31] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – mining discriminative components with random forests," in European Conference on Computer Vision,2014.

[32] F. Cholletet al., "Keras," https://keras.io, 2015.

[33] T. Kluyver, B. Ragan-Kelley, F. P´erez, B. E. Granger,M. Bussonnier, J. Frederic, K. Kelley, J. B. Hamrick,J. Grout, S. Corlayet al., Jupyter Notebooks-a publishing format for reproducible computational workflows., 2016, vol. 2016.

[34] P. Chen, B. B. Liao, G. Chen, and S. Zhang, "Under-standing and utilising deep neural networks trained with noisy labels," in International Conference on MachineLearning. PMLR, 2019, pp. 1062–107

**Shroog Alshomrani** received her B.S degree from Prince Sattam Bin Abdulaziz University, Riyadh, Saudi Arabia, in 2017. She is currently a student in the M.S artificial intelligence at Computer Science Department at Umm Al-Qura Uniersity, Makkah, Saudi Arabia, and expected to receive her degree in 2022.

**Lina Aljoudi** received her B.S degree from Umm Al-Qura University, Makkah, Saudi Arabia, in 2018. She is currently a student in the M.S artificial intelligence at Computer Science Department at Umm Al-Qura University, Makkah, Saudi Arabia, and expected to receive her degree in 2022.

**Banan Aljabri** received her B.S degree from Umm Al-Qura University, Makkah, Saudi Arabia, in 2019. She is currently a student in the M.S artificial intelligence at Computer Science Department at Umm Al-Qura University, Makkah, Saudi Arabia, and expected to receive her degree in 2022.

**Sarah Al-Shareef** received the B.S. degree in computer science from King Abdulaziz University, Jeddah, Saudi Arabia, in 2005 and the M.S. degree in advanced computer science from Sheffield University, Sheffield, United Kingdom, in 2009 and a PhD degree in computer science from Sheffield University, Sheffield, United Kingdom, in 2015. Currently, she works as Assistant Professor in Computer Science Department at Umm Al-Qura University, Makkah, Saudi Arabia. Also, she is a member of IEEE Computer Society, IEEE Young Professional, IEEE Signal Processing Society. Her research interests include speech and Arabic technologies and especially automatic speech recognition ad acoustic modelling.