

Stacking and Voting Ensemble Models for Improving Food Image Recognition

Suhas Suresh
Department of Computer Science
California State University
Northridge, CA, USA
suhas.suresh.737@my.csun.edu

Abhishek Verma
Department of Computer Science
California State University
Northridge, CA, USA
abhishek.verma@csun.edu

Abstract— Food image recognition has drawn much attention recently because of its potential to transform the food business by automating food identification and streamlining the ordering and delivery process. One of the significant challenges in developing a food image recognition system is the high variability in the appearance of food items. The primary objective of this research is to build a food image recognition system based on deep learning.

We propose two majority voting ensemble models, which outperform their state-of-the-art level-0 CNN models, EfficientNetV2S, InceptionV3, and Fusion - LSTM + InceptionV3. Furthermore, we propose seven meta-learner stacking models based on the three level-0 models. The best performing stacking model was the SVM linear stacking model and achieved 93.1% classification accuracy on the challenging Food-101 dataset. It improved upon the three level-0 models and the two voting ensemble models. MLP stacking model also improved upon the level-0 models and the voting ensemble models.

Keywords— deep learning, Food-101, InceptionV3, EfficientNetV2S, stacking, majority voting, meta-learner.

I. INTRODUCTION

Automated methods for food image recognition have grown in relevance with conceivable applications in nutrition analysis, meal journaling apps, nutritional inspection, and other areas [1]. Furthermore, leveraging technology to manage one's diet ties to the increased focus on observing a healthy lifestyle.

The last several years have witnessed a surge in the dissemination of food photos due to the widespread use of cell phones and social media. However, human evaluation and classification are severely hampered by the enormous quantity of food photos uploaded online and hence the need to design automated systems that could recognize food images with high accuracy and speed. Previously, a unique technique was employed to mine discriminative parts utilizing Random Forests. This allowed for the simultaneous mining of parts for all classes and the transfer of knowledge across them [25].

More recent studies have used Convolutional Neural Networks (CNN) to detect food in images [2]. A CNN is a multiple-layered neural network that has a special architecture that is intended to accurately predict the output by extracting progressively more sophisticated elements from the data at each layer. There is room for improvement in accuracy of the CNN models for food image recognition, especially when dealing

with images with complex backgrounds [3]. Some of the other challenges are high variability in the appearance of food items, occlusion, and different illumination conditions. There are few guarantees regarding the capacity of the model to generalize given the size of the training loss in today's highly overparameterized models, sharpness-aware minimization (SAM) optimizer was proposed to address this issue [26].

Our paper's main goal is to use deep learning techniques to create a food image recognition model that can categorize foods from images with high accuracy and speed. We propose seven stacking models and two voting ensemble models that improve upon their state-of-the-art level-0 models in terms of accuracy. Furthermore, we compare the runtime of various proposed models. The rest of the sections of this paper are structured as follows. Section II focuses on the prior relevant works while section III provides details of the dataset. Section IV covers the proposed models. Section V and VI discuss experiment setup and present the findings of the experiments respectively and section VII discusses conclusion and future work.

II. RELATED WORK

In recent years, there has been much interest in food image recognition research. For this objective, several deep learning models have been proposed, each with its unique advantages and disadvantages [4][23][24]. This section reviews a few of the related studies that have been done in food image recognition. The early studies on this subject relied on manually created features, such as color histograms and texture data, which were extracted from food images and then categorized using machine learning techniques. However, these approaches had difficulties in accurately extracting the intricate and varied features found in food images.

CNNs for food image recognition has been the center of research in recent years. One of the first such works was Kawano and Yanai's [2] deep CNN model for food image classification on the Food-101 dataset [5]. It used a modified version of the AlexNet [18] architecture and achieved a classification accuracy of 62.3%. Another architecture named VGGNet [6] achieved high accuracy on the Food-101 dataset. It used a number of 3x3 convolutional filter layers, followed by max pooling.

Residual network (ResNet) architecture was developed to address the issue of vanishing gradients during training and was first presented by He et al. [7]. On several benchmark datasets,



Figure 1. Sample images of several different food categories from the Food-101 dataset [5].

including Food-101 [5] and UECFood100 [8], it has been demonstrated that this architecture achieved good accuracy [2]. Another well-known CNN architecture, Inception, was first presented by Szegedy et al. [9]. It can capture both the local and global features in the input image since it has numerous layers of filters with different widths. On a number of benchmark datasets, Inception has demonstrated high accuracy [9]

Mixup [10] is a data augmentation method that was applied to the recognition of food images. Chen et al. [11] introduced a CNN-based model that combined local and global food image characteristics, leading to enhanced results in food recognition. Random Erasing [12] is another data augmentation method where a rectangular patch from the input image is randomly erased and replaced with random noise. Augmented input data used to train a self-supervised model achieved a Top-1 accuracy of 76.5%, matching the performance of the ResNet-50 model. It outperformed AlexNet with fewer labels, achieving a Top-5 accuracy of 85.8% when fine-tuned on just 1% of the labels [13].

Foret et al. [14] combined foreground objects with backdrop sceneries to create synthetic food images. They showed that by training CNN models on many combinations of actual and fake data, the accuracy of recognition was increased. Estimating portion sizes or calorie content is another facet of food image recognition.

In summary, deep learning approaches, particularly CNNs, have greatly improved the accuracy and robustness of food image recognition systems. Transfer learning, data augmentation, and multi-task learning have been employed to tackle challenges associated with limited annotated datasets and portion size estimation. While much progress has been made, there is a need for further improvement in developing more efficient and accurate food image recognition models.

III. DATASET DESCRIPTION

We perform experiments on the Food-101 [5] dataset, which has 101 distinct food classes. The dataset was first made public in 2014 and was developed by the Computer Vision Laboratory at ETH Zurich. Each food class has 1,000 images. There are a total of 101,000 images. The sample images from the dataset are shown in Figure 1.

The food images were manually collected from various online sources to ensure that they included a wide range of food categories, including fruits, vegetables, meats, cereals, sweets, etc. The culinary categories include more specialized categories like tiramisu, fish & chips, and baby back ribs in addition to general categories like pizza, sushi, hamburgers, and apple pie. The images are diverse sizes, aspect ratios, quality, and illumination levels. Some images have several different food items, and some are partially obscured by other things. The images are divided into training and validation. There are around 25,500 images in the validation set and around 75,500 images in the training set.

IV. RESEARCH METHODOLOGY

A. Data Pre-Processing

The success of the final model depends heavily on the data pre-processing stage. We resized all images to the same dimension to fit the input size of the deep learning model. The pixel intensity values are scaled between 0 and 1 to normalize the images. Normalizing the images speeds up learning and lessens the chance of gradients bursting or vanishing during training. We also performed batch normalization to speed up and stabilize the training of deep convolutional neural networks.

B. Convolutional Neural Networks (CNN)

A CNN is a deep learning method that decodes images by using a layer-by-layer neural network that has been specifically designed for the task. It basically consists of three types of layers, convolutional, pooling, and fully connected layers. A convolutional network's top layer is called the convolutional layer. The fully connected layers usually come last. With each additional layer, a CNN grows more intricate and can recognize more parts of the image. The first few layers focus on the most fundamental elements, like colors and borders. CNN starts to identify the item's larger pieces or characteristics as the visual input moves through the layers, eventually recognizing the target object. Regularization [29] and batch normalization [27][28][30] technique is utilized to stabilize the training. Dropout is added to the fully connected layers to reduce overfitting and speed up convergence. The transfer learning technique makes use of a model that has already been trained for classification and recognition to further train the model on a smaller domain specific dataset.

C. InceptionV3

The InceptionV3 [9] deep convolutional neural network performs image categorization tasks. The architecture comprises of connected layers that use several activation functions. The main principle of the InceptionV3 design is to employ several inception modules to make the network learn features or characteristics at various spatial scales. These modules consist of several parallel convolutional layers with various filter sizes, a pooling layer, and a concatenation of the outputs from the parallel layers.

The InceptionV3 model requires an input size of 299 x 299 pixels, we scale all images to that size. We processed the images

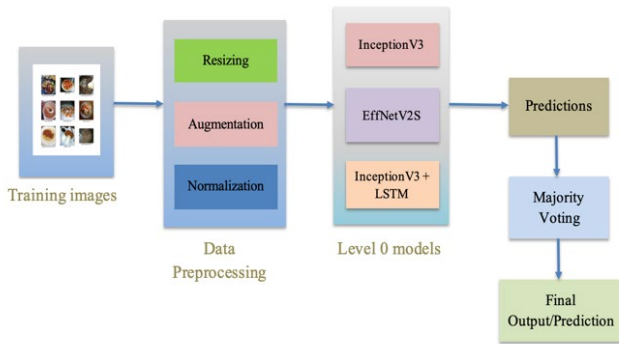


Figure 2. Architecture of the proposed majority voting ensemble model.

in batches during training using data generators. Data generators allow to train the model on enormous datasets that won't directly fit in memory while using less memory overall. We used pretrained weights and transfer learning to train the model on Food-101 dataset. The InceptionV3 model's top layers were partially unfrozen, and the entire model was retrained using a slower learning rate.

D. EfficientNetV2S

One of the most effective convolutional neural networks (CNN) models is EfficientNet [21]. EfficientNetV2 [22] takes one step further than EfficientNet. Its major objective was to increase parameter efficiency and training speed, which is several times smaller and far faster than the earlier version. We used the pretrained EfficientNetV2S [22] model to train on Food 101 dataset. This model is smaller, quicker to train, and scales well to growing data without sacrificing accuracy in comparison to prior variants. It provides a more effective and progressive learning technique that adaptively modifies both regularization and image size. This model uses both MBCConv and Fused-MBCConv layers, to optimize high accuracy and training speed.

E. Fusion - LSTM+ InceptionV3

This model was originally designed to perform multimodal classification of the same concept using both textual and visual input [20]. Two different models are trained and ensembled using early fusion technique in conjunction with ensemble strategy to implement a multi-modal classifier that outperforms state-of-the-art models on relevant datasets.

We used an adaptation of this model designed for image classification. It is based on a CNN model constructed on the architecture of InceptionV3. The last layer comprises 128 units, which is a long short term memory (LSTM) layer, whereas the other parallel final layer, which is a dense layer, comprises 128 units as in the Inception model. The outputs of those layers are passed into the concatenation layer. A 256-unit dense layer and a rectified linear unit (ReLU) activation function are added after concatenation. Lastly, the classification layer is a dense layer with 101 units and a SoftMax activation to work with the Food-101 dataset. This layer outputs the final classification results of the expected category of the given inputs.

F. Proposed Majority Voting Ensemble Models

Majority voting can be applied to enhance model performance and could produce results that are superior to those

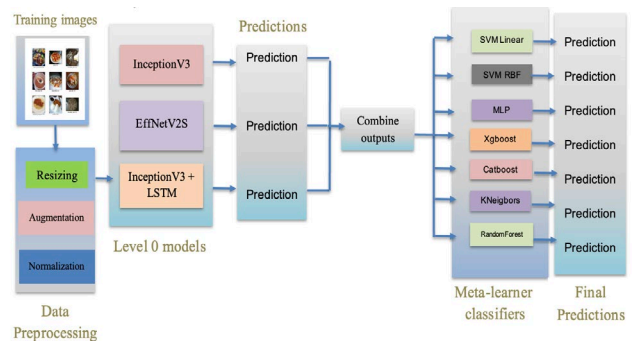


Figure 3. Architecture of the proposed meta-learner stacking models. Seven models are proposed based on seven different meta-learner classifiers.

of any one model utilized alone. The way a voting ensemble operates is by aggregating the results of forecasts from various models, the input is then assigned to the majority class. We propose soft majority voting and hard majority voting ensemble models that combine the predictions of level-0 models: InceptionV3, EfficientNetV2S, and Fusion-LSTM + InceptionV3. Figure 2 shows the architecture of the majority voting ensemble model.

ii. Proposed hard majority voting ensemble model

A hard voting model uses the mode of all the predictions from various level-0 models to classify input data. Depending on whether the weights assigned to the various level-0 models are equal or not, the majority vote is regarded differently. For our proposed model we assume equal weights.

Suppose there exists three level-0 models. The class labels for the predicted class for certain input data for each of the three level-0 models is $[1, 1, 0]$. If all the models have equal weights, the mode of predictions is chosen. As a result, class 1 is predicted for the specific input data because the mode of $[1, 1, 0]$ is 1.

ii. Proposed soft majority voting ensemble model

A soft voting model uses all the predictions from several level-0 models in terms of probabilities to classify input data. The weights assigned to each level-0 model are applied accordingly. Each level-0 model first assigns a probability to each class. The class with the highest overall probability is the ensemble's prediction. As an illustration, suppose there are three level-0 models, each trained to distinguish between images of pizza and pancake. The image is predicted to be of a pancake with a probability of 0.7 and a pizza with a probability of 0.3 by the first level-0 model, a pancake with a probability of 0.4 and a pizza with a probability of 0.6 by the second model, and a pancake with a probability of 0.6 and a pizza with a probability of 0.4 by the third model. The ensemble will predict the image is of a pancake.

G. Proposed Meta-Learner Stacking Models

We propose seven different meta-learner stacking models based on the three level-0 models discussed earlier. Figure 3 shows the architecture of the proposed meta-learner stacking models.

TABLE I. TOP-1 VALIDATION CLASSIFICATION ACCURACY PERCENT OF VARIOUS MODELS ON THE FOOD-101 DATASET

Models	Accuracy (%)
Level-0 Models	
InceptionV3	81.2
EfficientNetV2S	90.5
Fusion - LSTM + InceptionV3	92.2
Proposed Majority Voting Ensemble Models	
Hard Majority Voting Model	92.4
Soft Majority Voting Model	92.7
Proposed Meta-learner Stacking Models	
SVM RBF	90.2
KNeighbors	91.5
Random Forest	91.7
CatBoost	91.9
XGBoost	92.3
Multilayer Perceptron (MLP)	92.8
SVM Linear	93.1

An ensemble meta-learner is referred to as stacked generalization or simply stacking. It generalizes on data in a stacked fashion. One advantage of stacking is that it could outperform any single model in the ensemble by utilizing a variety of effective models. To determine how the predictions are combined from two or more underlying level-0 models, it employs a meta-learner classifier, which learns whether each member of the ensemble is to be trusted or not. In stacking, as

TABLE II. TOP-1 VALIDATION CLASSIFICATION ACCURACY PERCENT OF SOFT MAJORITY VOTING ENSEMBLE MODEL ON FOOD-101 DATASET (TOP 20 CLASSES SORTED IN DESCENDING ORDER OF ACCURACY)

Classes	Accuracy (%)
Chicken wings	100.0
Pizza	100.0
Chicken quesadilla	99.9
Hamburger	99.9
Donuts	99.9
Cheesecake	99.8
Garlic bread	99.8
Cup cakes	99.7
Chocolate cake	99.6
Sushi	99.5
Caesar salad	99.5
French fries	99.4
Breakfast burrito	99.3
Grilled cheese sandwich	99.1
Hot dog	99.0
Pancakes	98.9
Nachos	98.8
Ice cream	98.7
Fried rice	98.5
Apple pie	98.5

TABLE III. TOP-1 VALIDATION CLASSIFICATION ACCURACY PERCENT OF META-LEARNER STACKING SVM LINEAR MODEL ON FOOD-101 DATASET (TOP 20 CLASSES SORTED IN DESCENDING ORDER OF ACCURACY)

Classes	Accuracy (%)
Chicken wings	100.0
Chicken quesadilla	100.0
Pizza	100.0
Chocolate cake	100.0
Donuts	99.9
Sushi	99.9
Breakfast burrito	99.8
Cheesecake	99.8
French fries	99.6
Garlic bread	99.5
Hamburger	99.4
Grilled cheese sandwich	99.3
Cup cakes	99.3
Caesar salad	99.2
Fried rice	99.2
Nachos	99.2
Apple pie	99.2
Pancakes	99.1
Hot dog	99.1
Ice cream	99.0

opposed to bagging, the models are often unique and fitted to the same dataset. Stacking leverages an individual model to learn to integrate the results from level-0 models, as opposed to an order of models that adjusts the findings of previous models.

V. EXPERIMENT SETUP

In this section, we discuss hardware and software setup and machine learning model setup to train the deep learning models. The Food-101 dataset is split randomly with 75% and 25% of the images for training and validation, respectively.

A. Hardware and Software Setup

All the experiments in this research were carried out on an Apple MacBook Air with an 8-core CPU, four performance cores, four efficiency cores, a 16-core Neural Engine, and an eight GB unified memory. Some of the experiments were conducted on Google Colab. Python Jupyter notebook was used to implement the code and visualize the results. TensorFlow [15] and Keras [16] machine learning, deep learning, and Python libraries were used for implementation. TensorFlow [15] enables developers to experiment with novel optimizations and training algorithms.

B. Pre-processing and Data Augmentation Setup

Exploratory data analysis of the dataset's images uncovered certain issues. The images in the dataset range in size from 183 x 183 to more than 4,000 x 4,000. However, only 20% of the images are smaller than 400 x 400 or larger than 973 x 973, with 50% of the images falling between 190 x 190 and 852 x 852. We resized the images to 299 x 299. Random rotation was employed coupled with a random horizontal flip to augment the dataset.

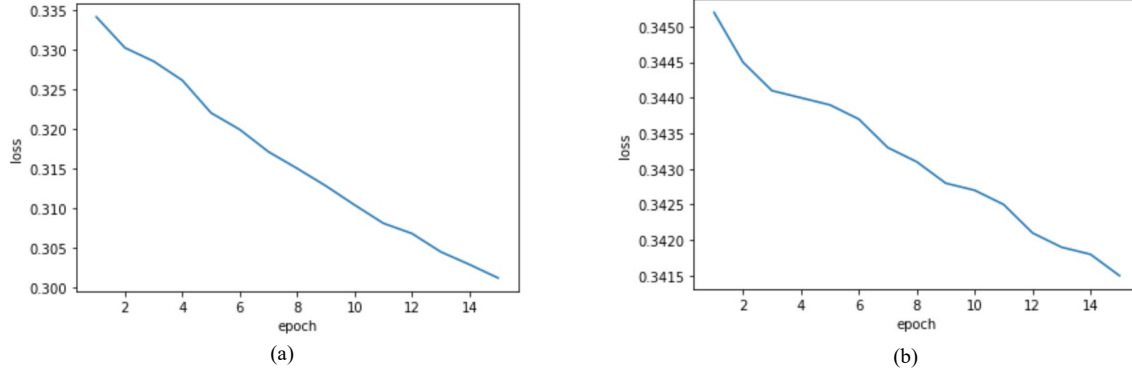


Figure 4. (a) Training and (b) validation loss plot for Meta-learning Stacking Multilayer Perceptron (MLP) model on the Food-101 dataset.

C. Transfer Learning and Setup of Level-0 Models

Transfer learning is a machine learning strategy where a learned model is used to train a new model for a different problem domain. Knowledge is transferred between domains using the pre-trained network. Training deeper neural networks is more challenging and multiple deep learning applications take advantage of transfer learning to train on smaller datasets.

Two pre-trained models, InceptionV3 and EfficientNetV2S, that are originally trained on the ImageNet dataset [19], are loaded for transfer learning. The Conv2D layer's input shape is (299, 299, 3) to match the size of the input image. The output is flattened to a 1D vector using the dense layer after the final max pooling 2D layer and is then passed through fully linked Dense layers with 128 neurons. There is a ReLU activation function in the dense layer. Two dropout layers with a rate of 0.2 are utilized, one before and one after the first dense layer, to avoid overfitting. Finally, the model is built using a categorical cross-entropy loss function. Stochastic gradient descent (SGD) optimizer is used, which is an adaptive optimization technique to find locally optimal weights. The model is tested using the validation set after being trained for 10 epochs with a batch size of 64. The model checkpoints for both models are saved in an hdf5 file to evaluate and visualize results.

D. Proposed Majority Voting Ensemble Models Setup

Voting is implemented via the Scikit-learn [17] Python machine learning package. The voting input is set to either hard or soft. The predictions from three different level-0 models are aggregated into an ensemble that casts votes. Each voting ensemble model is tested repeatedly with a 10-fold cross-validation method. This function receives an instance of the model and returns the scores from 10-fold cross-validation iterations. Next, each method's average performance is calculated.

TABLE IV. MODEL SIZE AND TRAINING TIME FOR LEVEL-0 MODELS ON FOOD-101 DATASET

Model	Parameter Count (million)	Size on Disk (MB)	Epochs	Total Training Time
InceptionV3	23.9M	92	10	7h 21m
EfficientNetV2S	21.6M	88	10	5h 2m
Fusion - LSTM + InceptionV3	110M	180	15	8h 3m

E. Proposed Meta-learner Stacking Models Setup

We used Scikit-learn library to implement stacking of three level-0 or base models. The final estimator provides the level-1 model or meta-model. Cross-validation is used to prepare the meta model. We use seven different meta learner classifiers at level-1 namely Support Vector Machine (SVM) Linear kernel, SVM Radial Basis Function kernel (RBF), KNeighbors, Random Forest, XGBoost, CatBoost, Multilayer Perceptron (MLP), and compare their results in the next section.

VI. RESULTS AND DISCUSSION

A. Results of Level-0 Models

All three level-0 models were trained on the ImageNet dataset and the pre-trained model is trained on the Food-101 using the SGD optimizer and the parameter values LR = 0.0001, batch size = 32, training ratio = 0.75, testing ratio = 0.25, and epochs = 10. Table I shows the Top-1 classification accuracy percent on the Food-101 dataset.

After training the InceptionV3 model for 10 epochs the validation accuracy is 81.2%. The EfficientNetV2S model was trained for 10 epochs and the validation accuracy is 90.5%. After training the Fusion-LSTM+InceptionV3 model for 15 epochs the validation accuracy is 92.2%, which is the highest of the three level-0 models.

From Table IV it could be observed that the EfficientNetV2S is quickest to train with fewest parameters and size on disk compared to the other two level-0 models. Fusion - LSTM+InceptionV3 has the highest accuracy, but it expectedly has more parameters, higher size on disk, and larger total training time.

TABLE V. MODEL SIZE AND TRAINING TIME FOR META-LEARNING STACKING MODELS ON FOOD-101 DATASET

Model	Size on Disk (MB)	Total Training Time
SVM RBF	68	6h 27m
KNeighbors	53	5h 45m
Random Forest	57	5h 13m
CatBoost	71	6h 11m
XGBoost	69	6h 02m
MLP	120	7h 57m
SVM Linear	77	6h 38m

B. Results of Proposed Majority Voting Ensemble Models

Table I shows the validation accuracies of hard voting and soft voting models as 92.4% and 92.7% respectively. Both voting models improved upon the level-0 models. As expected, soft majority voting gave slightly better results. The classification accuracy results of soft voting for the top 20 classes can be seen in Table II. Majority voting implementation is computationally inexpensive.

C. Results of Proposed Meta-learner Stacking Models

Table I shows the validation accuracies of seven different meta-learner stacking models. The best performing stacking model was trained with SVM linear at 93.1%. Both SVM linear and MLP models improved upon the voting models and level-0 models. XGBoost improved upon the all level-0 models and was at par with the voting models. The accuracy results of best performing meta-learner stacking SVM linear model for the top 20 classes can be seen in Table III. Figure 4 shows the training and validation loss of MLP stacking model across 15 epochs of training. Table V presents a comparison of the model size and training time for the seven meta-learning stacking models.

VII. CONCLUSION AND FUTURE WORK

In this paper we proposed two majority voting ensemble models, which outperformed their level-0 models. Furthermore, we proposed seven meta-learner stacking models based on the three level-0 models. The best performing stacking model was the SVM linear stacking model that improved upon the three level-0 models and the voting ensemble models. MLP stacking model also improved upon the level-0 models and the voting ensemble models. The models were trained on the challenging Food-101 dataset, which has high variability in terms of food images, varied illumination, and occlusion. Future work will focus on fusing larger transformer models on bigger datasets along with multi-class and multi-modal classification of food images, which is an active and challenging area of research.

REFERENCES

- [1] M. Wazumi, X. -H. Han, D. Ai and Y. -W. Chen, "Auto-recognition of food images using SPIN feature for Food-Log system," 6th Int. Conf. on Computer Sciences and Convergence Information Technology (ICCIT), Seogwipo, Korea (South), 2011, pp. 874-877.
- [2] K. Yanai and Y. Kawano, "Food image recognition using deep convolutional network with pre-training and fine-tuning", IEEE Int. Conf. on Multimedia & Expo Workshops (ICMEW), Turin, Italy, 2015.
- [3] J. Huang, L. Qu, R. Jia and B. Zhao, "O2U-Net: A Simple Noisy Label Detection Approach for Deep Neural Networks," 2019 IEEE/CVF Int. Con. on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 3325-3333.
- [4] S. Mezgec and B. K. Seljak, "Using Deep Learning for Food and Beverage Image Recognition," 2019 IEEE Int. Conf. on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 5149-5151.
- [5] Bossard, Lukas and Guillaumin, Matthieu and Van Gool, Luc. "Food-101 - Mining Discriminative Components with Random Forests", 2014, EECV (European Conf. on Computer Vision)
- [6] K. Simonyan. and A. Zisserman. (2014) "Very Deep Convolutional Networks for Large-Scale Image Recognition". arXiv preprint arXiv:1409.1556
- [7] Kaiming He & Zhang, Xiangyu & Ren, Shaoqing & Jian Sun. (2016). "Deep Residual Learning for Image Recognition". 770-778. 10.1109/CVPR.2016.90.
- [8] Matsuda, Y. and Hoashi, H. and Yanai, K., "Recognition of Multiple-Food Images by Detecting Candidate Regions", Proc. of IEEE Int. Conf. on Multimedia and Expo (ICME)," 2012.
- [9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, & A. Rabinovich. (2015). "Going deeper with convolutions". In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (pp. 1-9).
- [10] Z. Zhang, S. Xu, S. Cao, & S. Zhang. (2018). "Deep convolutional neural network with mixup for environmental sound classification". In Pattern Recognition and Computer Vision: First Chinese Conf. PRCV 2018, Guangzhou, China, November 23-26, 2018, Proc. Part II 1 (pp. 356-367). Springer Int. Publishing.
- [11] T. Chen, S. Kornblith, M. Norouzi and G.Hinton. "A simple framework for contrastive learning of visual representations". In Int. Conf. on Machine Learning, pp. 1597-1607. PMLR, 2020.
- [12] Z. Zhong, L. Zheng, G. Kang, S. Li, & Y. Yang. (2020, April). "Random erasing data augmentation". In Proc. of the AAAI (Association for the Advancement of Artificial Intelligence) Conf. on Artificial Intelligence (Vol. 34, No. 07, pp. 13001-13008).
- [13] E.D. Cubuk, B. Zoph, J. Shlens, and Le, "Q. V. Randaugment: Practical automated data augmentation with a reduced search space". In Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops, pp. 702-703, 2020.
- [14] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. "Sharpness-aware minimization for efficiently improving generalization". In 9th Int. Conf. on Learning Representations, ICLR, 2021.
- [15] Abadi, Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). TensorFlow: A system for large-scale machine learning. In 12th Symp. on Operating Systems Design and Implementation (pp. 265-283).
- [16] F. Chollet, & others. (2015). Keras. GitHub. Retrieved from <https://github.com/fchollet/keras>.
- [17] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- [18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Neural Information Processing Systems, Lake Tahoe, NV, 2012, pp. 1097-1105.
- [19] O. Russakovsky, J. Deng, H. Su, et al., "ImageNet Large Scale Visual Recognition Challenge," Int. J. of Computer Vision, 2015.
- [20] I. Gallo, G. Ria, N. Landro, and R. La Grassa, "Image and Text fusion for UPMC Food-101 using BERT and CNNs," Int. Conf. on Image and Vision Computing, New Zealand, Nov 2020, pages 1-6.
- [21] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," Proc. of the 36th Int. Conf. on Machine Learning, PMLR 97:6105-6114, 2019.
- [22] M. Tan and Q. Le, "EfficientNetV2: Smaller Models and Faster Training," Proc. of Int. Conf. on Machine Learning, 2021.
- [23] J. Zheng, Z. J. Wang and X. Ji, "Supapixel-based image recognition for food images," 2016 IEEE Canadian Conf. on Electrical and Computer Engineering (CCECE), Vancouver, BC, Canada, 2016, pp. 1-4.
- [24] W. Zhang, D. Zhao, W. Gong, Z. Li, Q. Lu and S. Yang, "Food Image Recognition with Convolutional Neural Networks," 2015 IEEE 12th Int. Conf. on Ubiquitous Intelligence and Computing, Beijing, China, 2015, pp. 690-693.
- [25] L. Bossard, M. Guillaumin, L. Van Gool, "Food-101 - Mining Discriminative Components with Random Forests". In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision - ECCV 2014.
- [26] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. "Sharpness-aware minimization for efficiently improving generalization". In 9th Int. Conf. on Learning Representations, ICLR, 2021.
- [27] S. Ioffe and C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In ICML (Int. Conf. on Machine Learning, 2015.
- [28] I. Gitman, and B. Ginsburg, "Comparison of batch normalization and weight normalization algorithms for the large-scale image classification," arXiv preprint arXiv:1709.08145, 2017.
- [29] P. Luo, X. Wang, W. Shao, and Z. Peng, "Towards understanding regularization in batch normalization," arXiv preprint arXiv:1809.00846, 2018.
- [30] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2018. arXiv preprint arXiv:1607.06450, 2016.