Food Image Classification Using Deep Learning

Zackary A. Kinsman, Igor Vouffo

Mentors: Dr. Nalinda Kulathunga, Dr. Yunjiao Wang and Dr. Daniel Vrinceanu.

Summer Program in data Science - 2019, Department of Mathematics, Texas Southern University, Houston, TX 77004.

1. Abstract

In data science, machine learning is widely used to build predictive models. In the present time, mathematical and computational tools have been successfully used to implement and execute deep neural networks and algorithms especially in image classification. In this work we investigate the mathematical optimization techniques used for the loss minimization and apply deep learning tools for the classification of food images into 11 classes. We explain how the Gradient Descent algorithm works and use the faster Stochastic Gradient Descent loss for the loss minimization. For the feature extraction we use the basic image data pre-processing techniques together with Convolutional Neural Networks and briefly explain their operation. We monitor the overfitting of the model using loss and accuracy learning curves. For the evaluation of the model we use the accuracy score and results are presented in the form of confusion matrix. Transfer learning was used to enhance the accuracy and reduce the training time. The performance of the models were tested using different loss functions, optimizers, and pre-trained models. We were able to achieve 81% test accuracy by using the bottleneck features extracted from InceptionV3 pre-trained model.

2. Method (Steps in brief)

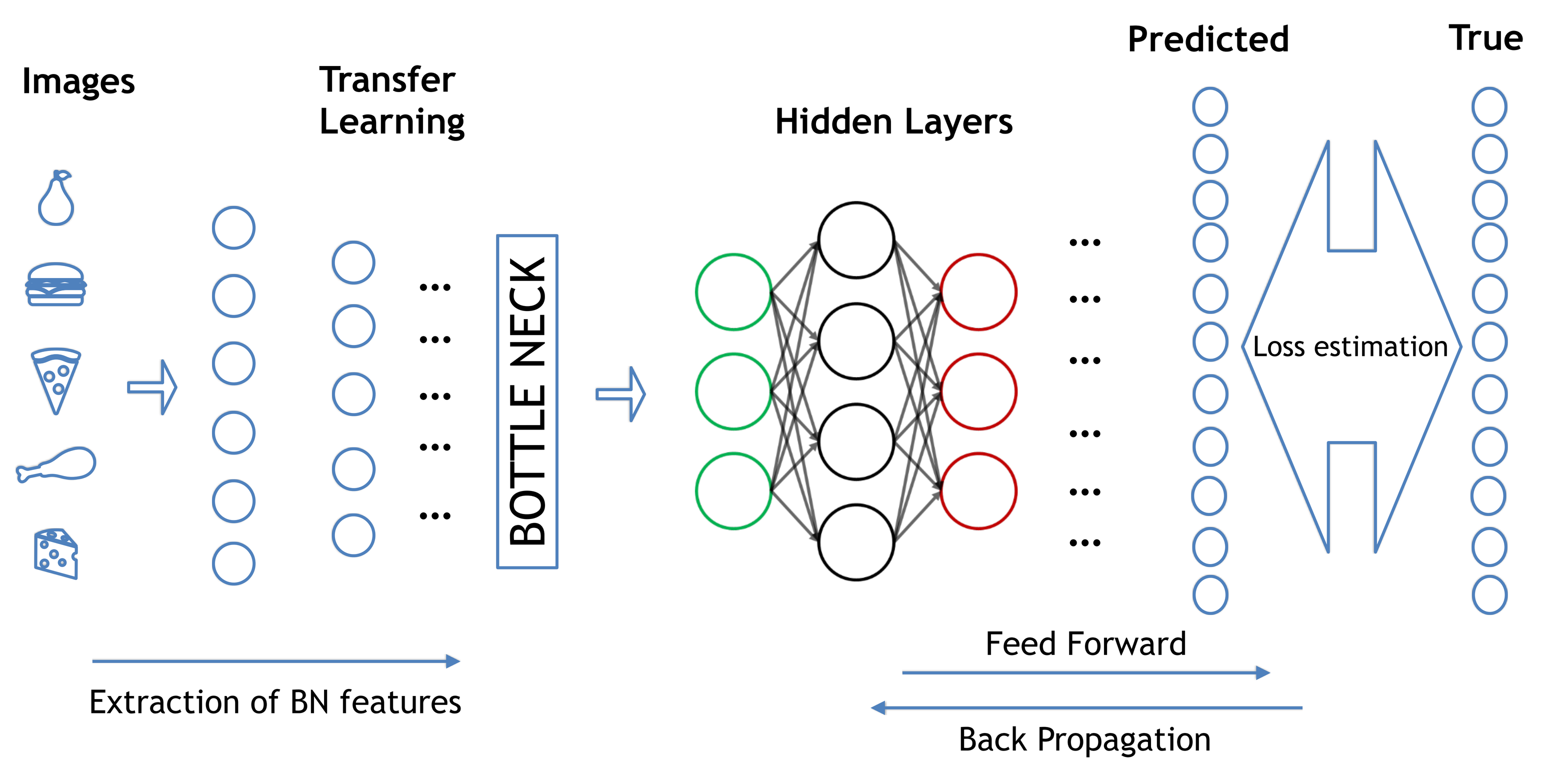
Data set: Food-11 [1] (3300 training and 550 validation images)

Classes: Bread, Dairy Products, Dessert, Egg, Fried Food, Meat, Noodles Pasta, Rice, Seafood, Soup, Vegetables or fruits.

Transfer learning models: VGG16 [9], InceptionV3 [4].

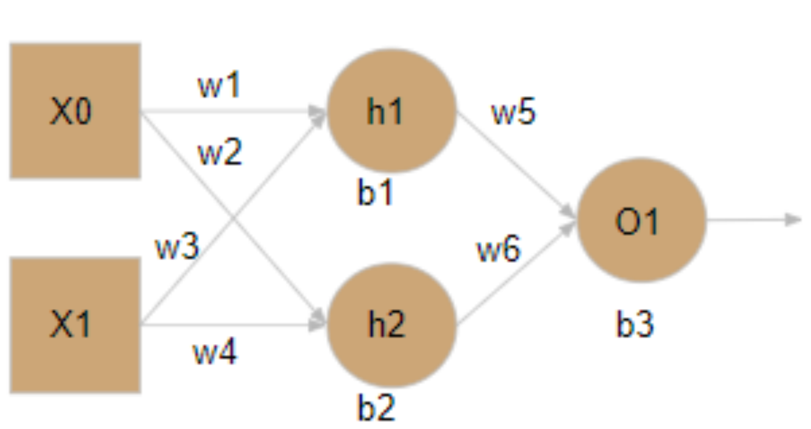
Optimizers: Mean Squared Error (MSE), Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (ADAM) [2].

Evaluation: Accuracy confusion matrix.



3. Neural Networks

Feed Forward: Images are converted into tensors. In each node (neuron), input vectors get multiplied by the corresponding weights (w) and passed through an activation function (f).



Hidden layers

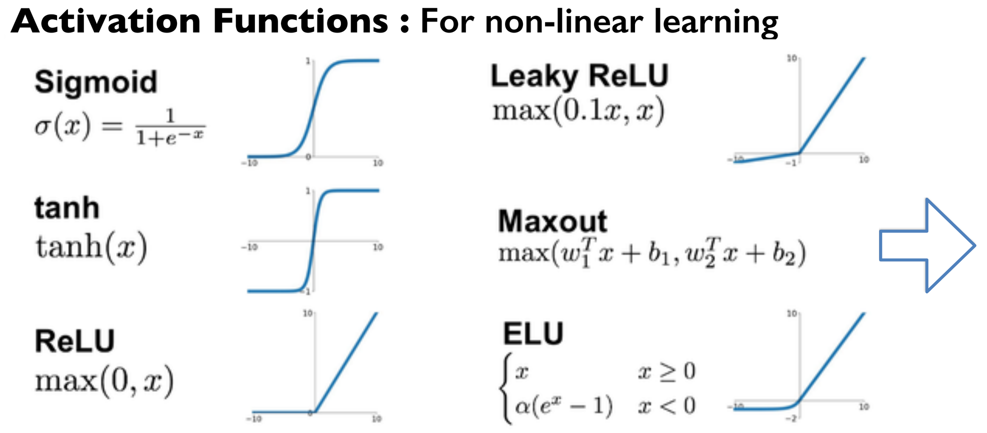
$$h1 = f(w1 * X0 + w3 * X1 + b1)$$

$$h2 = f(w2 * X0 + w4 * X1 + b2)$$

Output layer

$$O1 = f\big(h1 * w5 + h2 * w6 + b3\big)$$

Activation Functions : For non-linear learning

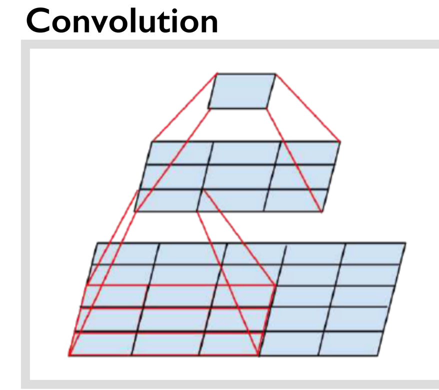


Activation functions are used to impose the non linear learning. Without them, all the layers of a deep neural network will break down to a one single layer. This reduces the learning capability and always represent the predictions as a linear combination of feature space which may lead to low testing accuracy.

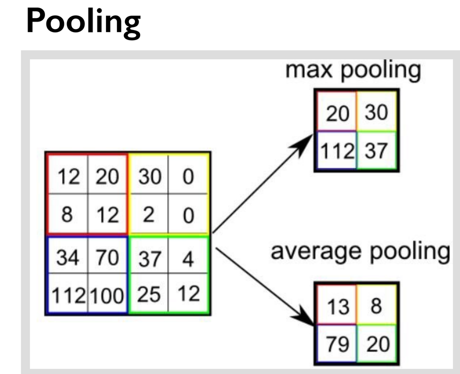
Back Propagation : In each training epoch, the output from the feed forward process is compared with the actual label and loss is calculated. Famous loss functions: Mean Squared Error, Mean Absolute Error, Categorical Cross Entropy. Then the optimization is done using the gradient of error function as explained in section 4. Finally the weights are updated until the expected minimum loss is achieved.

4. Convolutional Neural Networks (CNNs)

Convolution

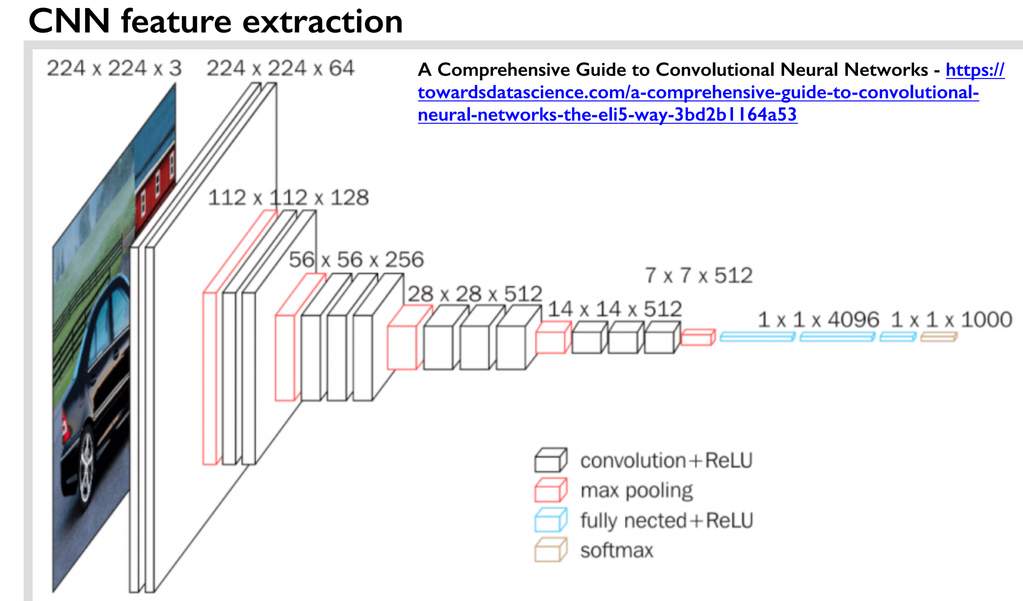


Pooling



Convolution process is a way of extracting impotent features from images. Filters with pre-defied sizes and weights are used to scan through strides along all directions in the input shape. Pooling operations reduces the image dimensionality and make the classification easier. CNNs are followed by a fully connected layer for the purpose of clarification/regression.

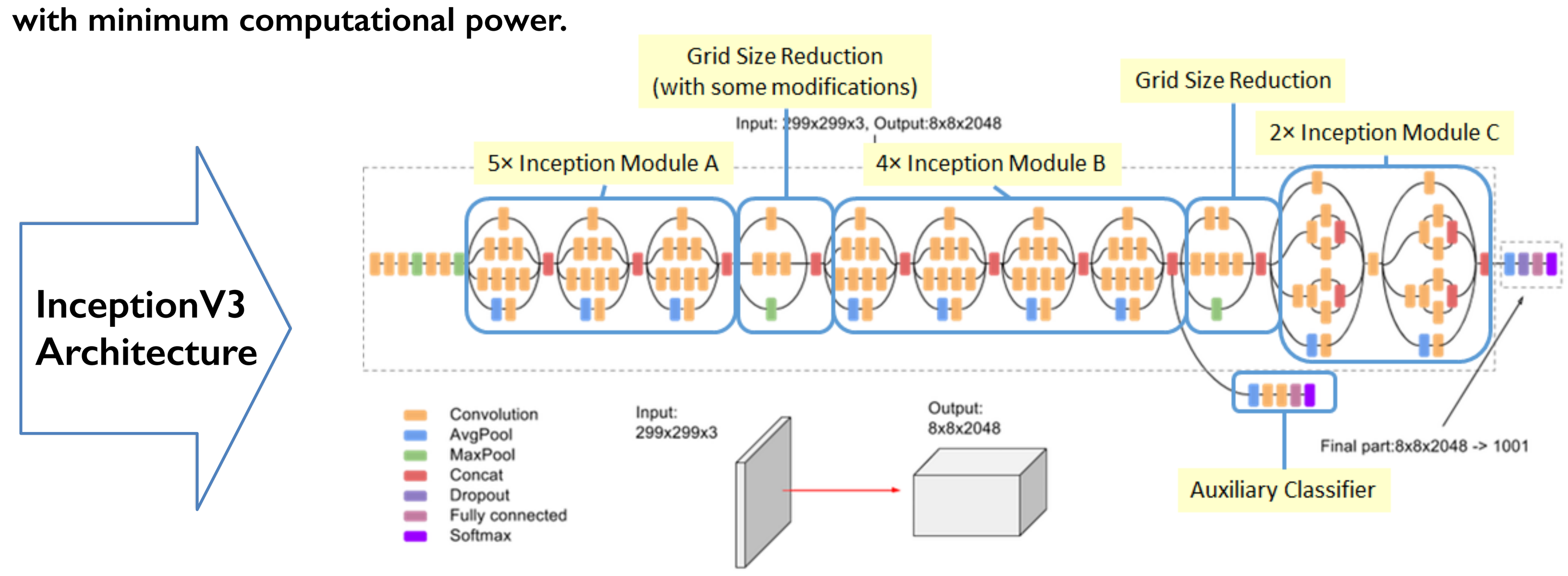
CNN feature extraction



A Comprehensive Guide to Convolutional Neural Networks - https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

5. Transfer Learning and Bottleneck Features

The process of using a pre-trained model to improve your accuracy is called transfer learning. To classify our images, we used two different pre-trained models, InceptionV3 [4] and VGG16 [9]. VGG16 forms the bottle neck layer by passing the image through multiple convolution layers with a 3 x 3 receptive field with a stride of one pixel. The InceptionV3 model is a 42-layer deep learning that uses the same filter size as the VGG16 but has 7 million parameters as opposed to VGG with over 180 million. Using the weights from these models and attaching them to our model using a bottleneck feature layer enables us to get higher accuracy with minimum computational power.



Review: Inception-v3 – 1st Runner Up (Image Classification) in ILSVRC 2015 - https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c

4. Loss Minimization Methods

In gradient descent algorithms, models can be trained by minimizing the Mean Square Error (MSE). Consider the matrix representation of the problem: Y = θX, where Y represents labels, X represents features and θ represents the weights. The MSE(θ) and the gradients are defined as;

$$MSE(X,\theta) = \frac{1}{m}\sum_{i=0}^{m}(y_{pred}^{(i)} - y^{(i)})^2 = \frac{1}{m}\sum_{i=0}^{m}(\theta^T . x^{(i)} - y^{(i)})^2$$

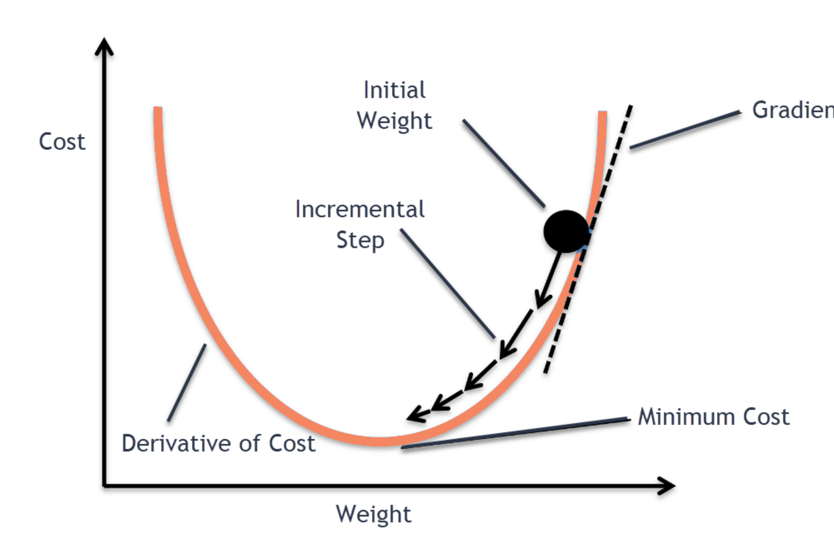
$$\frac{\partial}{\partial\theta_j}MSE(X,\theta) = \frac{2}{m}\sum_{i=0}^{m}(\theta^T . x^{(i)} - y^{(i)})x_j^{(i)}$$

Here, m stands for number of data instances, *i* stands for the index related to length of the data set and *j* stands for the number of features (considering *j* = 0 for the bias term). Weights will be updated in each iteration so that the MSE(θ) reaches a global minimum with a rate η, which is also known as the learning rate.

$$\theta^{(next)} = \theta^{(now)} - \eta\nabla_\theta MSE(\theta)$$

SGD is a faster mode of the gradient descent where the gradient is calculated in each iteration from randomly selected instances of the data set.

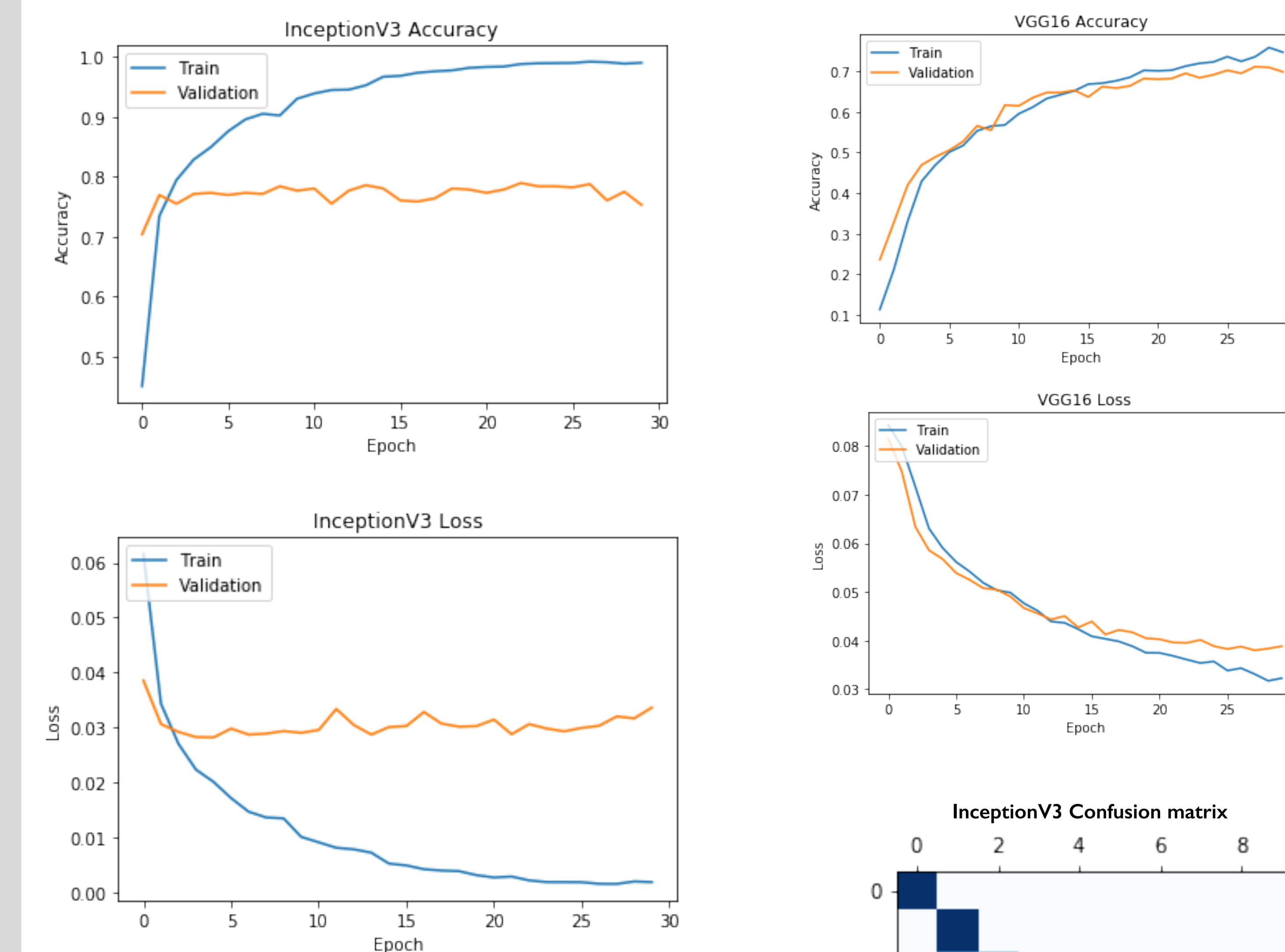
ADAM stands for adaptive moment estimator. This is an improved version of Gradient Descent algorithm where the optimization is done with the help of SGD (with momentum) and RMSprop[6] . This has been a very successful in achieving higher accuracies for models trained using Deep Neural Nets.



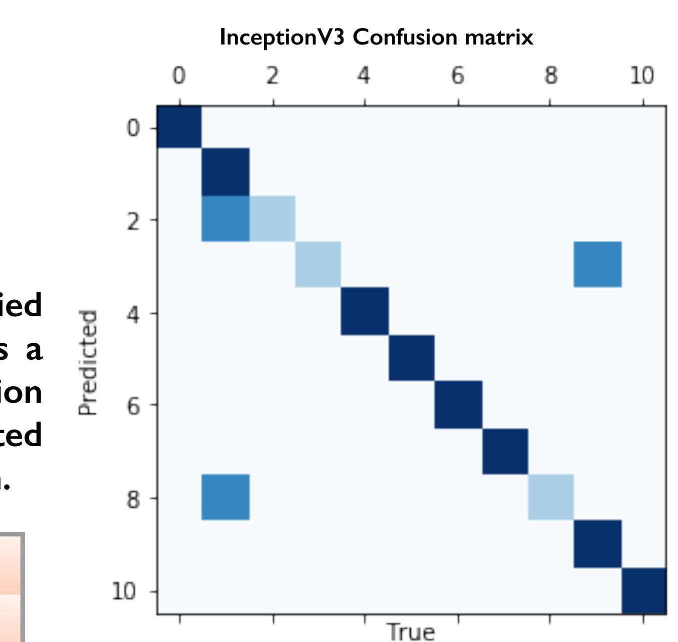
Stochastic vs Batch Gradient Descent - https://medium.com/@divakar_239/stochastic-vs-batch-gradient-descent-8820568eada1

7. Results (Learning Curves/Confusion Matrix)

To evaluate the accuracy and loss that achieved with each pre-trained model's bottleneck features, we used four different combinations of loss functions and optimizers: MSE with ADAM, MSE with SGD, categorical cross-entropy with ADAM, and categorical cross-entropy with SGD. Below are the learning curves and confusion matrices from training our neural network with the bottleneck features from InceptionV3 and VGG16 respectively. The results shown are obtained using MSE and ADAM optimizers. The presented confusion matrix is based on the predictions using InceptionV3.



This confusion matrix shows how our model classified the testing images. A light-colored box represents a lower occurrence than a dark color. The confusion matrix is designed to show the model's predicted classification of an image vs the labeled classification.



Pre-Tr. Model	Validation Loss	Test Accuracy
InceptionV3	0.0282	81.8%
VGG16	0.0386	51.5%

8. Discussion

In this work we investigated about different approaches to build a predictive model to identify different food images and classify them in to 11 different classes using deep learning. We presented a short discussion about the popular optimization technique; Gradient Descent used in loss minimization in the training of deep neural networks. In this work we have used an improved version of gradient descent called ADAM optimizer for the model training. Among the different optimizers and loss functions used, we have found that Mean Squared Error (MSE) with ADAM results in the best test accuracy. For a faster classification process we used transfer learning with pre-trained models. VGG16[9] and InceptionV3 [10] are the two pre-trained models used for the extraction of bottleneck features. We have found that the InceptionV3 bottleneck features used with our model outperform the VGG16 bottleneck features with every combination of loss and optimization functions we used. We have reached a test accuracy of 81.8% with MSE as the loss function and ADAM as the optimizer in the food classification project.

Reference :

Websites:

1. Food Image Dataset - https://mmspg.epfl.ch/downloads/food-image-datasets/
2. Adam – latest trends in deep learning optimization. - https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c
3. Machine Learning for Beginners: An Introduction to Neural Networks - https://victorzhou.com/blog/intro-to-neural-networks/
4. Review: Inception-v3 – 1st Runner Up (Image Classification) in ILSVRC 2015 - https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c
5. A Comprehensive Guide to Convolutional Neural Networks - https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
6. An overview of gradient descent optimization - http://ruder.io/optimizing-gradient-descent/index.html#adam
7. Stochastic vs Batch Gradient Descent - https://medium.com/@divakar_239/stochastic-vs-batch-gradient-descent-8820568eada1
8. What is a Convolutional Neural Network? - https://hub.packtpub.com/what-is-a-convolutional-neural-network-cnn-video/
9. VGG 16 - https://neurohive.io/en/popular-networks/vgg16/

Articles:

10. Very Deep Convolutional Networks for Large –Scale Image Recognition, Karen Simonyan et. al.
11. Rethinking the Inception Architecture for Computer Vision, Christian Szegedy et. al.

Acknowledgment :

This work has been supported by National Science Foundation through an Excellence in Research award (CNS-1831980).