Combining Deep Learning Techniques and Support Vector Machine (SVM) for Vegetable Image Classification

Zhiqi Bei, Wenhan Liu, Zhuxian Ding, Yue Huang
SYDE 522, Option B
Systems Design Department
University of Waterloo, Canada
{zbei,z73ding,y743huan,w288liu}@uwaterloo.ca

***Abstract*—Vegetable images are a valuable source of data and contain abundant information for pattern recognition. The machine learning techniques for vegetable classification can be used at varies occasions to recognize the types of vegetables purchased by consumers and for further analysis. In this paper, three widely utilized algorithms - convolutional neural network (CNN), ResNet, and support vector machine (SVM) are combined for vegetable recognition and classification. In the designed model, CNN and ResNet serves as feature extractors, whereas SVM serves as a classifier. The testing data has shown that CNN+softmax can reach to 99.62% and ResNet-50+softmax also with 99.80%, while CNN+SVM model can only reach the accuracy of 93.93% and ResNet-50+SVM of 97.70%. All experiments provide sufficient results for this study.**

***Index Terms*—CNN, SVM, ResNet, image processing**

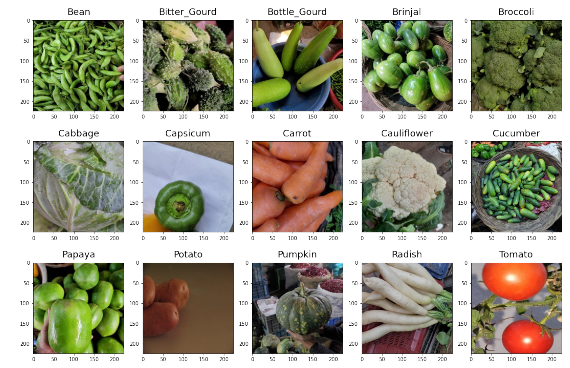


Fig. 1. Vegetable species. There are totally 15 classes used for classification

I. Introduction

Vegetables are a mainstay in most people's diets all over the world, and there are around 100,000 distinct vegetable species on the planet. Since various vegetables have a lot of similarities, it may be difficult for a market shopper to identify a vegetable. In this case, the creation of a vegetable classifier will provide clarity to those vegetable species and prevent high time-consuming works for customers [1]. Designing a classification system for vegetable identification is a challenging task to the substantial intra- and interclass differences in vegetable varieties (shown in Figure 1). Also, these images contain complex background such as degrees of illumination, disruption by other objects, and the views from different angles [2]. In addition, images with large size (224×224) have increased the difficulty of image processing. With those challenges, the implementation of high quality pattern recognition techniques is essential for classifying vegetables.

Numerous methods have been developed by researchers for object recognition, and one of the major achievement is convolutional neural network (CNN) [3]. As part of artificial neural network (ANN) in deep learning, CNN are widely being implemented in the computer vision field to enhance the system's initial design in order to attain improved accuracy. The main advantage of CNN is feature extraction, which is due to the information contained by each pixel that can be distinguished.

Normally, softmax function is used in the final fully connected layer for classification. This function returns the probability for a data point belonging to each individual class [4]. However, there are recently studies conducted that compares softmax classifier to an alternative - the support vector machine (SVM) [5], [6]. As a binary classification method, SVM [7] was proposed by Corinna and Vladimir in 1995 and it provides several advantages in solving nonlinear and high dimensional classification problems. SVM may also apply for overfitting problems and extract more features from images [8].

For data sets with complicated structures, deep network structure is required for more sophisticated feature pattern extraction. However, as the number of layers of the model increases, the network structure becomes more complex, resulting in two typical problems of gradient vanishing and gradient explosion. As one of the enhanced models, ResNet [9] has solved these two problems in deep structure to a certain extent, by improving the way of simple stacking layers that causing the degradation problem of deep network. The combination of the techniques mentioned is expected to perform a high quality vegetable image recognition.

In this paper, the vegetable recognition and classification are approached by comparing softmax and SVM in both CNN and ResNet models. Sections 2 and 3 will provide background information on those techniques as well as a quick overview of the methodology. Section 4 will describe how the experiments were carried out as well as the results, and Section 5 will sum up the paper.

II. Background

A. Convolutional Neural Network - CNN

The proposal of CNN was first in 1980, by Kunihiko Fukushima in his *Neocognitron* [10]. The process of forward propagation of CNN achieved by convolution blocks and fully connected layers is only multiple linear and non-linear functions. What essentially makes CNN a 'learnable' network is back propagation, which was firstly presented by Seppo Linnainmaa in 1970 [11]. Back propagation allows the weight of each neuron to update by adding the 'gradient' generated during calculating partial derivatives until the local optimal solution is obtained. In convolution part, CNN provides 'filters' with size for input images and extract and learn each feature (pixel) by scanning the image with preset stride and padding. The processed images from the convolutional layer are called 'convolution maps' [3]. Then an activation function is applied for the outputs (usually a ReLU function) to activate or deactivate the neurons [12]. A pooling layer is inserted before the next convolutional layer, which refers to approximating the outputs by aggregating nearby values and reducing the size of the convolution maps [3]. After being processed by convolutional layers and pooling layers, the input images are forwarded to the fully connected layers and classified by using classifier. An example of the structure of CNN is shown in Figure 2.

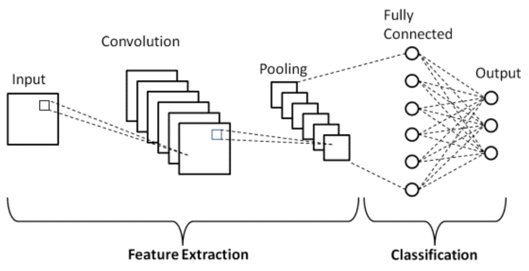


Fig. 2. CNN structure. The structure contains a convolution block, and fully connected layers for classification

B. Support Vector Machine - SVM

The core concept of SVM is margin, which is a linear classifier with the largest interval defined on the feature space. However, the utilization of 'kernel trick' makes SVM become a substantially nonlinear classifier, which refers to mapping the non-linearly divisible samples from the original space to a higher dimensional space so that the samples are linearly divisible in the new space [13]. The learning strategy of SVM is interval maximization, which can be formalized as a problem of solving a convex quadratic programming (shown in Figure 3). There are two general approaches for SVM to achieve multiclass classification: OVR (one versus rest) and OVO (one versus one) [14]. For OVR, *k* SVMs are trained for *k* categories, and the *j*th SVM is used to determine whether any data belongs to category *j* or non-*j*. While OVO allows *k*(k-1)/2* SVMs to be trained to determine which of two specific categories in k any data belongs to [14].

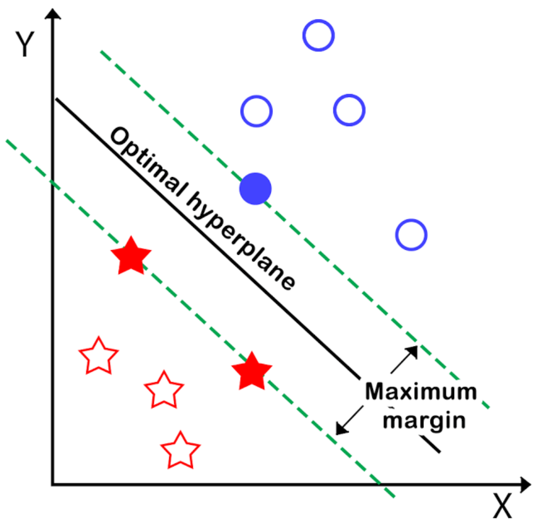


Fig. 3. Principle of SVM. The approach is to solve the maximum margin between support vectors

C. Residual Neural Network - ResNet

ResNet [9] was proposed in 2015 as an enhanced model of CNN. ResNet adds directly connection to the weight layers, called 'identity' [9]. In addition, with the idea of residual learning, the network can directly pass the output information to the previous layers by identity connection during the back propagation, thus to protect the integrity of the information [9]. The structure of ResNet is shown in Figure 4. Conventional CNN maps *x* to *y=F(x)* output when adding layers. In ResNet, the output of each layer is *y=F(x)+x*. Instead of learning the representation of the output feature y directly, this is where *y-x* is learned. If the representation of the original model is learned, then all the parameters of F(x) are set to 0, which refers to a constant mapping. *F(x)=y-x* is also called the residual, and it is easier-https://www.overleaf.com/project/625480b6a1624ba39eca5e02 to learn the residual than the full mapping form.

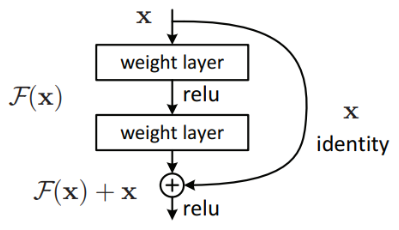


Fig. 4. ResNet structure. The idea with identity connection and residual to prevent gradient vanishing/explosion in deep structure

III. Method

A. Machine Learning Library

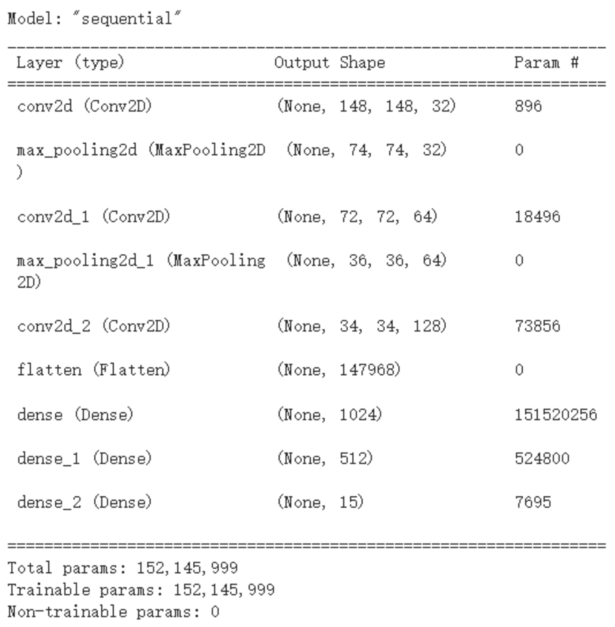
The deep learning techniques in this paper were implemented using the Pytorch framework, which is based on the Torch library.

B. CNN Structure

For CNN design, we implemented our own CNN as Fig.5. shows. For our CNN architecture, the convolutional layers (Conv) consist of 3×3 size filters to scan through the input images with 1 stride. The pooling layers (Pool) apply max pooling to reduce the size of convolution maps. ReLU function introduces non-linearity for CNN to learn the parameters. Fully connected layers (FC) are implemented for final classification. The entire structure is shown below:

1) Input: 150×150×3
2) Conv: 32 channels, 3×3 kernel_size, stride of 1
3) ReLU activation function
4) MaxPool: 2×2 kernel_size, stride of 2
5) Conv: 64 channels, 3×3 kernel_size, stride of 1
6) ReLU activation function
7) MaxPool: 2×2 kernel_size, stride of 2
8) Conv: 128 channels, 3×3 kernel_size, stride of 1
9) Flatten Layer
10) Dense: 1024 neurons
11) Dense: 512 neurons
12) Dense: 15 output classes

For experimental comparison, instead of using the conventional softmax function at the CNN's last fully connected layer, the SVM is implemented.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 148, 148, 32)      896

 max_pooling2d (MaxPooling2D  (None, 74, 74, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 72, 72, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 36, 36, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 34, 34, 128)       73856

 flatten (Flatten)           (None, 147968)            0

 dense (Dense)               (None, 1024)              151520256

 dense_1 (Dense)             (None, 512)               524800

 dense_2 (Dense)             (None, 15)                7695

=================================================================
Total params: 152,145,999
Trainable params: 152,145,999
Non-trainable params: 0
```

Fig. 5. Architecture of the CNN model. With the output shape from each layer

C. SVM

In SVM, the equation for the hyperplane can be written in the following form:

$$w^T x + b = 0 \tag{1}$$

Each hyperplane corresponds to a margin, and the goal of SVM is to find the hyperplane that corresponds to the largest value among all margins. This is an optimization problem and the objective function can be written as:

$$\arg\max_{w,b}((y(w^T x + b))\frac{1}{||w||}) \tag{2}$$

where y denotes the label of the data point and it's -1 or 1. The above problem can be simplified as:

$$\arg\max(\frac{1}{||w||}) \quad s.t.\ y(w^T x + b) - 1 \geq 0 \tag{3}$$

For the convenience of later calculations, the objective function is equivalently replaced by:

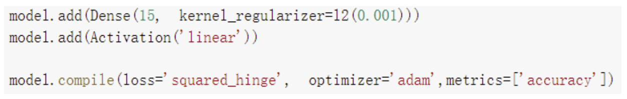
$$\min \frac{1}{2}||w||^2 \tag{4}$$

This minimization problem represents the primal form of the hard margin SVM. However, for soft-margin SVM, we need to combine the minimization objective with a loss function to get better classification results. The loss function chosen in this project for multi-classification is the **squared hinge** loss [15], which defined as:

$$L(y, \hat{y}) = \sum_{i=0}^{N} \left(max\left(0, 1 - y_i \cdot \hat{y}_i\right)^2 \right) \tag{5}$$

where y denotes the true label of the data which is -1 or 1, and $\hat{y}$ denotes the predicted value. Thus, the value of the squared hinge loss is equal to 0 when the true and the predicted labels are the same or when $\hat{y} \geq 1$, and the value is quadratically increasing with the error when the true and the predicted labels are different or when $\hat{y} < 1$ [16].

As it's shown in Figure 6, in the output layer of the CNN and ResNet model, we convert original softmax activation into SVM. To do that, we implement *kernel_regularizer* as the regularizer and use $L2$ norm inside. Then, *linear* is passed as activation function and squared hinge is used as the loss function during the compiling for SVM.



```
model.add(Dense(15,  kernel_regularizer=l2(0.001)))
model.add(Activation('linear'))

model.compile(loss='squared_hinge',  optimizer='adam',metrics=['accuracy'])
```

Fig. 6. SVM classifier. Using squared hinge loss function

D. ResNet-50

ResNet-50 [17] is used in this paper. Basically, ResNet-50 contains two blocks: convolution block and identity block. The convolution block consists of convolutional layers, batchnorm layers for normalization, and ReLU function. The input and output dimensions are not the same, thus each convolution block cannot be linked in series consecutively, and the role was originally to change the dimension of the image. The identity block provides the same input and output dimensions. Still, both softmax and SVM are used in the final fully connected layer. The structure is shown in Fig. 7.

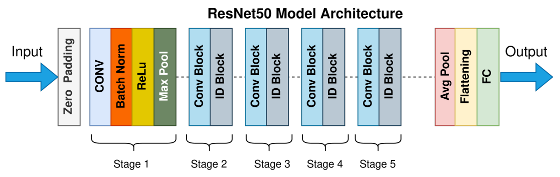


Fig. 7. The structure of ResNet-50. Consists of convolution block and identity block with total 50 layers.

Due to the existence of residual, the learning from the shallow layer to the deep layer can be represented as:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i) \tag{6}$$

Then according to the chain rule, the gradient of the back propagation process can be found as follows:

$$\frac{\partial loss}{\partial x_l} = \frac{\partial loss}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial loss}{\partial x_L}\left(1 + \frac{\partial}{\partial x_L}\sum_{i=l}^{L-1} F(x_i, W_i)\right) \tag{7}$$

The first factor of Eq. 7 represents the gradient of the loss function arriving at L. The 1 in parentheses indicates the short-cut mechanism can propagate the gradient without loss, and the other residual gradient needs to go through the layer with weights. Even if the residual gradient is small, the presence of 1 will not cause the gradient to disappear, which means residual learning will be easier. The residual network is a combination of many parallel subnetworks, and the whole residual network is equivalent to a voting system (ensemble).

IV. Results & Analysis

A. The Dataset

The vegetable image dataset [18], [19] the project used contains 15 types of vegetables with total 21000 images. Each class contains 1400 images of size 224×224. The dataset split 70% for training, 15% for validation, and 15% for testing (shown in Table I).

TABLE I
Dataset distribution for vegetable images. 70% for training, 15% for validation, and 15% for testing.

Dataset	Number
Training	15000
Validation	3000
Testing	3000

B. Experiments

For this network architecture, softmax and SVM are used separately as the final classifier layer for both CNN and ResNet-50 models. To test how these two different layers could affect the network classification, for both CNN and Resnet50 models, completely same model are used for the two comparisons between softmax and SVM. The only difference between the two comparisons are the final layers of the model. For both softmax and SVM experiments, the models are trained for only 5 epochs, and the results are already remarkable.

C. Results

1) CNN:

According to Fig. 8, during the training process, the CNN+SVM model has a better performance than the CNN+softmax model in every way, it has lower training and validation loss along with higher training and validation accuracy. However, as Table II shows, the final test accuracy for CNN+SVM is still slightly worse than the CNN+softmax model.

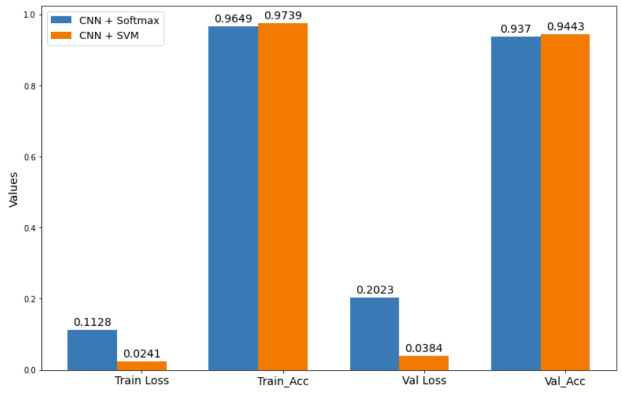


Fig. 8. Results of CNN. Detailed Comparison of training loss, training accuracy, validation loss and validation accuracy between the CNN+softmax (blue) model and CNN+SVM (orange) model.

TABLE II
The accuracy of each model(%).

Model	CNN+ Softmax	CNN+ SVM	ResNet50+ Softmax	ResNet50+ SVM
Training	96.49	97.39	99.72	98.24
Validation	93.70	94.43	99.90	87.87
Testing	99.62	93.93	99.80	97.70

2) ResNet-50:

It can be seen Fig. 9, during the training process, the ResNet-50+softmax model has a better performance than the ResNet-50+SVM model, which is the opposite of the above CNN results. It has lower training and validation loss along with higher training and validation accuracy. Besides, as Table II shows, the final test accuracy for ResNet-50+softmax is still better than the ResNet-50+SVM model.

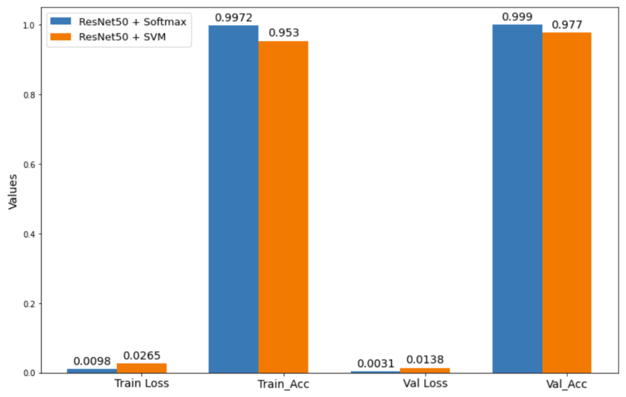


Fig. 9. Results of ResNet-50. Detailed Comparison of training loss, training accuracy, validation loss and validation accuracy between the ResNet-50+softmax (blue) model and ResNet-50+SVM (orange) model.

3) *Comparison between CNN and ResNet-50:*

The training and validation accuracy of the four models for 5 epochs is shown in Fig. 10. Both CNN and ResNet-50 do good performance on both training and validation datasets. The accuracies on these two datasets gradually increase with the number of training epochs and they are so close and higher than 93%, indicating that no overfitting or underfitting happened during the processes. Thus, the CNN and ResNet-50 models implemented are practical and efficient on the vegetable image dataset.

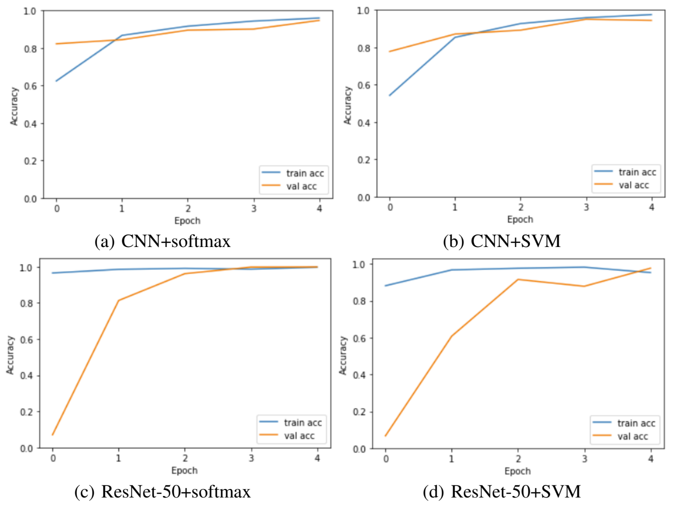


Fig. 10. Training and validation accuracy of the four models for 5 epochs. (a) CNN+softmax, (b) CNN+SVM, (c) ResNet-50+softmax and (d) ResNet-50+SVM. The blue curves represent the training accuracy and the orange ones represent the validation accuracy.

Meanwhile, with the same classifier layer (both with softmax or SVM), Fig. 10 shows that ResNet-50 always achieves the accuracy higher than 80% in the first epoch during the training process, while CNN can only reach the accuracy of 60%. On the contrary, during the validation process, CNN can achieve the accuracy higher than 70%, while the accuracy of ResNet-50 is very low. However, the accuracy of ResNet-50 on validation dataset increases sharply on the first 3 epochs and then reaches to a high value. The reason for this phenomenon may be that ResNet-50 has deeper architecture and more complex than CNN, which leads to more epochs needed to train the parameters to avoid overfitting.

Since CNN and ResNet-50 have different architectures, the parameters and training and testing time of these two models are also different, which is shown in Table III. It can be seen that CNN has more parameters than ResNet-50, but less training and testing time.

TABLE III
PARAMETERS AND EXPERIMENT TIME OF CNN AND RESNET50

Model	CNN	ResNet50
Parameters	152,145,999	26,218,383
Train time / min	20	33
Test time / min	1.5	0.5

V. CONCLUSIONS

For the CNN architecture, both experiments with softmax and SVM as final layer end up having good classification performance. The test accuracies are both high. However, the CNN model with softmax as final classifier layer has a slightly better test accuracy on this dataset. With the figure shown, although the CNN model with SVM as final layer outperforms the CNN model with softmax as final layer during both training and validation process, the final test accuracy shows that the CNN model with softmax as final layer is better in this case.

Similar to CNN, the performance of ResNet-50 with softmax and SVM on this vegetable are resulting a high accuracy. But even if they all have good performance, it can be seen from TABLE II that whether it is the training set, validation set or the test set, the accuracy of ResNet-50 model with softmax as the classification layer is slightly higher than using SVM as the classification layer.

So, for the reason why the model with softmax as classification has better testing accuracy, our assumption was that due to the characteristic of the dataset, the classification task required is a multiclass classification, and for the model with SVM, the activation method for the final layer was 'linear'. Hence, we think this might be the reason that why the model with SVM didn't perform as good as the model with softmax. And when it comes to the case where the classification task required was binary classification, we suspect the SVM model might outperform the softmax model.

References

[1] M Israk Ahmed, Shahriyar Mahmud Mamun, and Asif Uz Zaman Asif. Dcnn-based vegetable image classification using transfer learning: A comparative study. In *2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pages 235–243. IEEE, 2021.

[2] Khurram Hameed, Douglas Chai, and Alexander Rassau. A comprehensive review of fruit and vegetable classification techniques. *Image and Vision Computing*, 80:24–44, 2018.

[3] S. Zayen N. Jmour and A. Abdelkrim. Convolutional neural networks for image classification. In *2018 international conference on advanced systems and electric technologies (IC_ASET)*, pages 397–402. IEEE, 2018.

[4] Amr Ahmed, Kai Yu, Wei Xu, Yihong Gong, and Eric Xing. Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks. In *European Conference on Computer Vision*, pages 69–82. Springer, 2008.

[5] Xiao-Xiao Niu and Ching Y Suen. A novel hybrid cnn–svm classifier for recognizing handwritten digits. *Pattern Recognition*, 45(4):1318–1325, 2012.

[6] Abien Fred Agarap. An architecture combining convolutional neural network (cnn) and support vector machine (svm) for image classification. *arXiv preprint arXiv:1712.03541*, 2017.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] Shanshan Guo, Shiyu Chen, and Yanjie Li. Face recognition based on convolutional neural network and support vector machine. In *2016 IEEE International conference on Information and Automation (ICIA)*, pages 1787–1792. IEEE, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[11] Jürgen Schmidhuber. Who invented backpropagation. *More in [DL2]*, 2014.

[12] Y-Lan Boureau, Francis Bach, Yann LeCun, and Jean Ponce. Learning mid-level features for recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2559–2566. IEEE, 2010.

[13] Martin Hofmann. Support vector machines-kernels and the kernel trick. *Notes*, 26(3):1–16, 2006.

[14] Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2011.

[15] Ching-Pei Lee and Chih-Jen Lin. A study on l2-loss (squared hinge-loss) multiclass svm. *Neural computation*, 25(5):1302–1323, 2013.

[16] Guibiao Xu, Zheng Cao, Bao-Gang Hu, and Jose C Principe. Robust support vector machines based on the rescaled hinge loss function. *Pattern Recognition*, 63:139–148, 2017.

[17] Dhananjay Theckedath and RR Sedamkar. Detecting affect states using vgg16, resnet50 and se-resnet50 networks. *SN Computer Science*, 1(2):1–7, 2020.

[18] M. Israk Ahmed, Shahriyar Mamun, and Asif Asif. Dcnn-based vegetable image classification using transfer learning: A comparative study. pages 235–243, 05 2021.

[19] M Israk Ahmed. 'vegetable image dataset', 2021. https://www.kaggle.com/misrakahmed/vegetable-image-dataset.