[Team 2] Project C2: Leaf Wilting

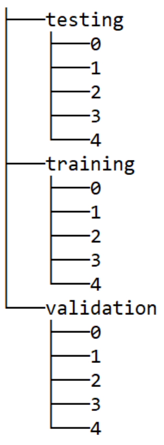
Mahmoud Abdelkhalek (maabdelk), Sam Messick (shmessic), Mason Shuler (mashuler)

I. Methodology

In [1], the authors survey 19 studies that rely on Convolutional Neural Networks (CNNs) to automatically identify crop diseases. The authors also provide a comparison of training times and accuracies for different CNN architectures in Figure 4 in the same paper. Based on this, AlexNet, ResNet34, and SqueezeNet1.1 were initially chosen for evaluation. The PyTorch deep learning framework was used to train, validate, and test these 3 models because of its native implementation of these architectures, as shown in [2]. The 3 networks were then combined using a majority voting rule to assess whether they performed better as a single classifier, an approach known as ”bootstrap aggregating” or ”bagging” for short. Also, an additional 3 networks were combined with the original 3 networks to assess whether a total of 6 networks would perform better as as a single classifier. Ultimately, however, the previous approaches suffered from the same problem: overfitting. The final design that performed the best was the individual AlexNet, whose architecture is shown in Figure 1. The ”AdaptiveAvgPooling2D” layer performs conventional average pooling, with the exception that the kernel size and other pooling parameters are derived from the specified output image size.

II. Model Training and Selection

A. Model Training



```
├──testing
│   ├──0
│   ├──1
│   ├──2
│   ├──3
│   └──4
├──training
│   ├──0
│   ├──1
│   ├──2
│   ├──3
│   └──4
└──validation
    ├──0
    ├──1
    ├──2
    ├──3
    └──4
```

Fig. 2.

The original TrainData-C2 dataset was split into a 80-10-10 training-validation-testing set, where each of the training, validation, and testing sets contained each of the 5 classes, as shown in Figure 2. Since classes 1,2,3, and 4 were under-represented compared to class 0, they were initially augmented so that all classes would be represented equally. For example, `training/0` currently contains 390 images, while `training/2` currently contains 104 images. Therefore, the images in `training/2` were augmented to contain a total of $104 \times 4 = 416$ images.

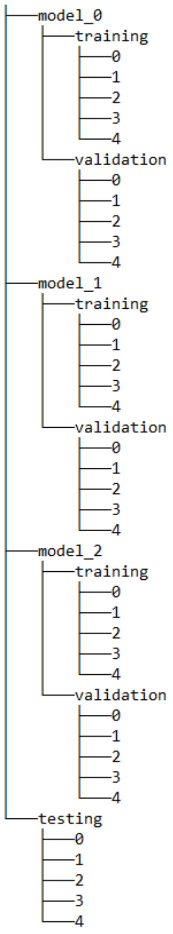
Initially, AlexNet, ResNet34, and SqueezeNet1.1 were each individually trained on the `training` set and validated on the `validation` set for 50 epochs, using a learning rate of 0.0001 and a batch size of 16. Their respective learning curves are shown in figures 3, 4, and 5. Finally, they were each tested on the `testing` set. Their respective testing accuracies were 82.39%, 86.76%, and 85.11%. Next, the 3 networks were combined using a majority voting rule and tested on the same `testing` set, such that the mode of the 3 predictions of the networks for each test image was chosen as the final decision. The final testing accuracy was 87.27%, which was only incrementally better than the best individual testing accuracy.

TABLE I
Class Histograms for AlexNet, SqueezeNet, and ResNet34

Epoch No.	AlexNet	SqueezeNet	ResNet34
5	[40,77,29,1,53]	[19,9,52,67,53]	[35 65 21 32 47]
15	[27,108,10,1,54]	[48,27,51,28,46]	[36 60 17 33 54]
30	[22,116,6,1,55]	[40,71,28,22,39]	[42 65 14 26 53]
50	[11,128,5,1,55]	[33,98,13,7,49]	[34 66 11 30 59]

Additionally, the original TrainData-C2 was evenly split into three (figure 6) and six (not shown) disjoint subsets and then augmented, and three (AlexNet, ResNet34, and SqueezeNet1.1) and six (AlexNet, ResNet34, SqueezeNet1.1, DenseNet169, Inception_v3, and VGG13) corresponding networks were trained for 50 epochs and validated individually on each of the augmented subsets. The group of three and group of six trained networks were then combined using a majority voting rule and tested on the same `testing` set. It was observed that the group of three networks performed better than the group of six networks, which could be attributed to the lack of diversity in the six disjoint subsets of the TrainData-C2 set.

It was also observed, however, that when all aforementioned approaches were tested on the un-labelled TestData set, they all suffered from overfitting, which was detected by analyzing the histogram of predictions made by the networks. These histograms are shown in table I. Note that the histograms are in the format of [A,B,C,D,E], where A is the number of predictions for class 0, B is the number of predictions for class 1, etc. It can be seen that by epoch 50, all networks were not able to accurately predict in classes 2 and 3. This also could be inferred from the learning curves in figures 3, 4, and 5.



```
├──model_0
│   ├──training
│   │   ├──0
│   │   ├──1
│   │   ├──2
│   │   ├──3
│   │   └──4
│   └──validation
│       ├──0
│       ├──1
│       ├──2
│       ├──3
│       └──4
├──model_1
│   ├──training
│   │   ├──0
│   │   ├──1
│   │   ├──2
│   │   ├──3
│   │   └──4
│   └──validation
│       ├──0
│       ├──1
│       ├──2
│       ├──3
│       └──4
├──model_2
│   ├──training
│   │   ├──0
│   │   ├──1
│   │   ├──2
│   │   ├──3
│   │   └──4
│   └──validation
│       ├──0
│       ├──1
│       ├──2
│       ├──3
│       └──4
└──testing
    ├──0
    ├──1
    ├──2
    ├──3
    └──4
```

Fig. 6.

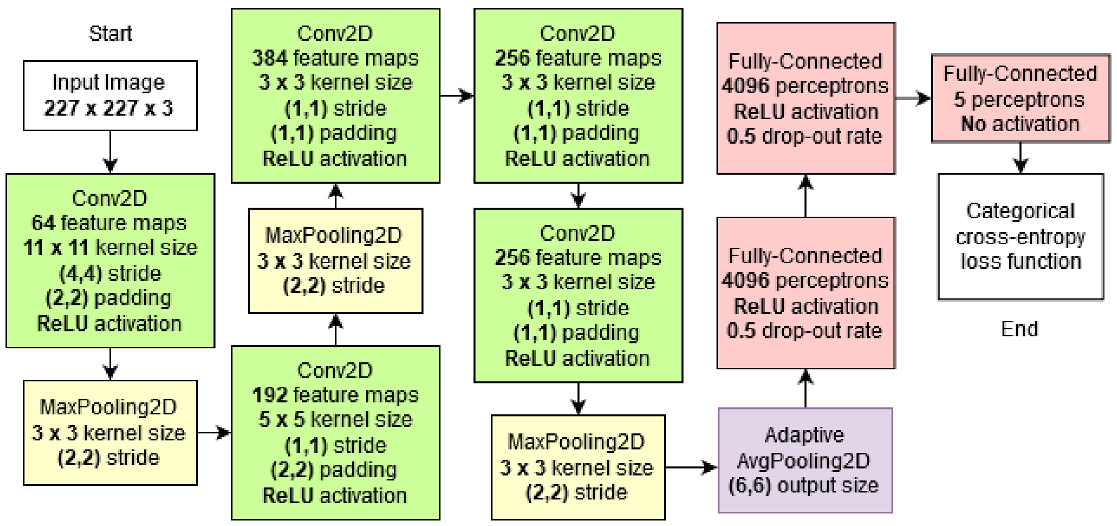


Fig. 1. Chosen AlexNet architecture

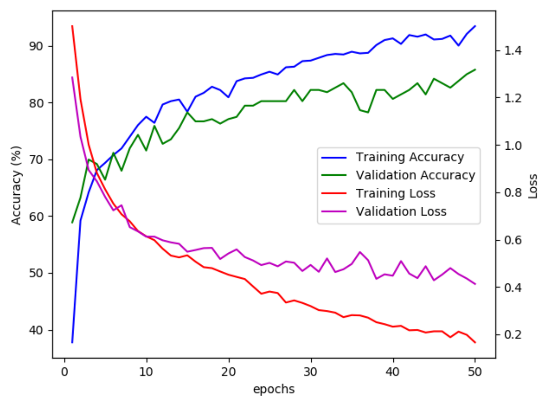


Fig. 3. AlexNet Learning Curves

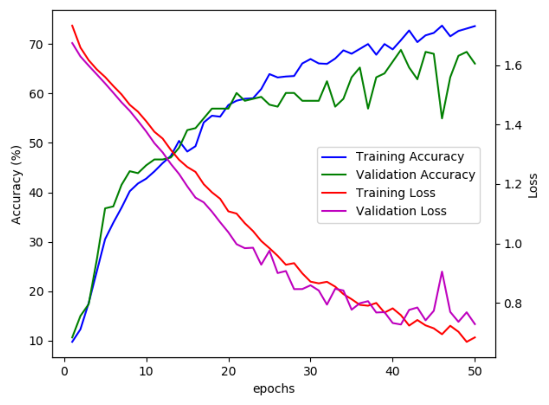


Fig. 4. SqueezeNet Learning Curves

B. Model Selection

To alleviate this, AlexNet was trained for 6 epochs on the un-augmented dataset shown in figure 2. Instead of augmenting the dataset, weights were assigned to each class in the

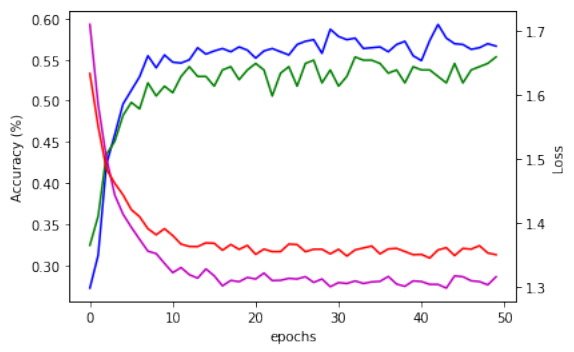


Fig. 5. ResNet34 Learning Curves

TABLE II
AlexNet Training Procedure with a Batch Size of 8

Epoch No.	Learning Rate	Train. Acc	Val. Acc	Histogram
1	0.0001	33.76%	56.92%	-
2	0.0001	51.76%	59.68%	[36, 81, 21, 13, 49]
3	0.00001	56.56%	59.68%	-
4	0.00001	57.05%	60.08%	[41, 82, 12, 10, 55]
5	0.000001	57.83%	60.08%	-
6	0.000001	55.87%	60.87%	[41, 82, 12, 10, 55]

categorical cross-entropy loss function. For example, class 2 was assigned a weight of $390/104 = 3.75$. Additionally, the initial learning rate was gradually decreased across epochs using a learning rate scheduler and the histogram of predictions made by the 3 nets on the TestData set was monitored every 2 epochs for different starting learning rates, decay rates, and batch sizes to check if overfitting occurred, as shown in tables II and III.

In table II, the learning rate decayed too rapidly, which meant that the network stopped updating its weights after each

TABLE III
ALEXNET TRAINING PROCEDURE WITH A BATCH SIZE OF 16

Epoch No.	Learning Rate	Train. Acc	Val. Acc	Histogram
1	0.00005	29.55%	41.90%	-
2	0.00005	45.11%	56.13%	[42, 72, 7, 8, 71]
3	0.00002	52.94%	58.89%	-
4	0.00002	58.22%	60.87%	[46, 73, 6, 12, 63]
5	0.000008	59.00%	62.03%	-
6	0.000008	59.39%	62.49%	[47, 75, 5, 13, 60]

TABLE IV
FINAL ALEXNET TRAINING PROCEDURE WITH A BATCH SIZE OF 16

Epoch No.	Learning Rate	Train. Acc	Val. Acc	Histogram
1	0.00005	30.33%	51.78%	-
2	0.00005	45.69%	60.08%	[60, 51, 2, 18, 69]
3	0.00005	53.23%	60.87%	-
4	0.000025	55.97%	63.64%	[56, 62, 3, 14, 65]
5	0.000025	60.96%	64.43%	-
6	0.000025	59.39%	67.59%	[57, 62, 10, 12, 59]

epoch. In table III, the class histograms were not as balanced as they could be, which again could be attributed to the low learning rate.

III. EVALUATION

The final training procedure that performed the best on the TestData set is shown in table IV. This model was also tested on the `testing` set shown in figure 2. The overall testing accuracy was 71.43%. Additional performance metrics are shown in table V.

TABLE V
ALEXNET WITHOUT WEATHER-DATA METRICS

Class No.	Precision	Recall	F_1 Score	Accuracy
0	0.755	0.755	0.755	0.755
1	0.576	0.594	0.585	0.594
2	0.667	0.462	0.545	0.462
3	0.588	0.769	0.667	0.769
4	1	0.947	0.973	0.947
Average	0.717	0.705	0.705	0.705

IV. EXTRA CREDIT: USING WEATHER DATA

A. Model Specification

The Adaptive AvgPooling2D layer in figure 1 outputs 256 features maps with a size of 6×6. Normally, these feature maps would be flattened to form a $256 \times 6 \times 6 = 9216$-dimensional input feature vector to the first fully-connected layer. However, to accommodate for the weather data, for every input image to AlexNet, the associated 12-dimensional feature vector containing its weather data is concatenated with the 9216-dimensional feature vector to form a $9216+12 = 9228$-dimensional feature vector, as shown in figure 7 [3].

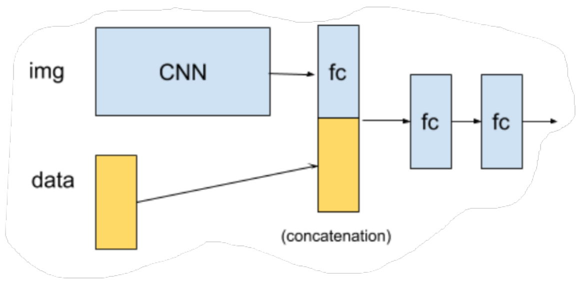


Fig. 7. Concatenation Method

Additionally, because the 12-dimensional feature vector containing the weather data contained values in the range of $[-5, 10^6]$, it was normalized to the range $[-1, 1]$ using the `torch.nn.functional.normalize()` function before concatenation to avoid vanishing and exploding gradients. More precisely, given a 12-dimensional feature vector $\mathbf{v}$, the normalized feature vector $\mathbf{v}_n$ is:

$$\mathbf{v}_n = \frac{\mathbf{v}}{\max(\|\mathbf{v}\|_2, 10^{-12})} \tag{1}$$

Where $\|\mathbf{v}\|_2$ is the Euclidean norm of $\mathbf{v}$ and 10^{-12} is used to avoid dividing by 0.

However, since the concatenation procedure led to the re-initialization of the weights of one of the fully-connected layers, this layer needed to be re-trained. Initially, the weights in the convolutional layers of AlexNet were frozen and the fully-connected layers were only trained using a learning rate of 0.00001, a learning rate decay rate of 3 epochs, and a decay factor of 0.8. As shown in figure 8, the learning rate was too low for the weights in the fully-connected layers to vary.

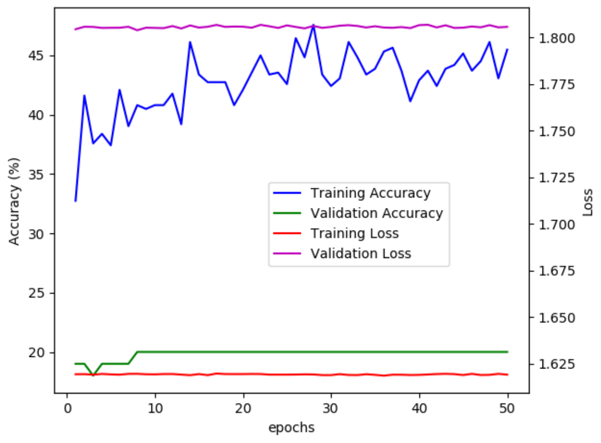


Fig. 8. AlexNet learning curve with a low learning rate

Next, the weights in the convolutional layers of AlexNet were un-frozen and the entire network was trained using a learning rate of 0.01, a learning rate decay rate of 3 epochs, and a decay factor of 0.2. As shown in figure 9, the learning rate was too high and the network eventually overfit to the training data. The learning rate decay rate and factor were not high enough to prevent overfitting.

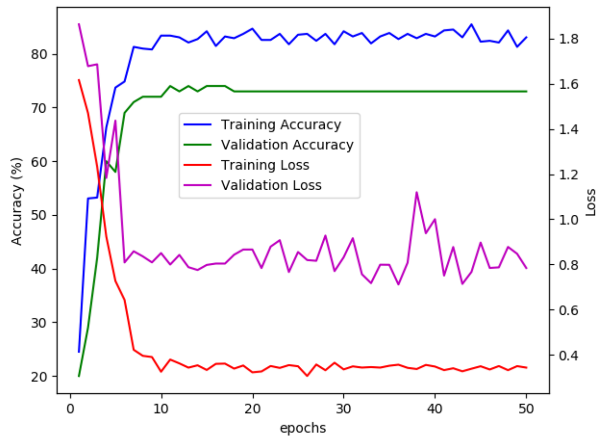


Fig. 9. AlexNet learning curve with a high learning rate

B. Evaluation

In the best-performing configuration, the convolutional layers of AlexNet were frozen and the fully-connected layers were trained using a learning rate of 0.005, a learning rate decay rate of 5 epochs, and a decay factor of 0.5, as shown in figure 10.

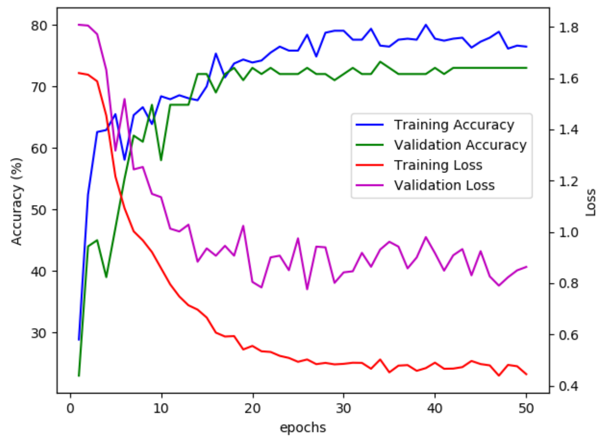


Fig. 10. Final AlexNet learning curves

When this model was tested on the testing set, it achieved a testing accuracy of 75%. Additional performance metrics are shown in table VI.

Comparing table V to table VI, it can be seen that AlexNet with the weather data slightly outerperforms AlexNet without the weather data. However, it is important to note that AlexNet without the weather data was trained on different data than the AlexNet with the weather data.

Finally, to check if the weather data alone can be used for correct classification, a Support Vector Machine (SVM) was

TABLE VI
ALEXNET WITH WEATHER-DATA METRICS

Class No.	Precision	Recall	F_1 Score	Accuracy
0	0.68	0.85	0.76	0.85
1	0.75	0.75	0.75	0.75
2	0.80	0.6	0.69	0.6
3	0.65	0.85	0.74	0.85
4	1	0.7	0.82	0.7
Average	0.776	0.75	0.752	0.75

trained using the `scikit-learn` Python package on the 620 12-dimensional training vectors containing the weather data. 4 different kernel functions were used: Linear, Radial Basis Function (RBF), Polynomial, and Sigmoid. Additionally, two main regularization parameter (C) values were used: 0.5 and 1. Finally, 2 different pre-processing techniques were performed on the data before training: min-max scaling and standardization. In min-max scaling, the features in all training vectors are scaled from a range of [-5,10^6] to [-1,1]. In standardization, the sample means and variances of each feature in all training vectors are used to transform them into zero-mean and unit variance feature vectors. Through experimentation, it was determined that min-max scaling and a 3^{rd} order polynomial kernel function with $C = 1$ achieved the highest testing accuracy of 70%. Additional performance metrics for this configuration are shown in table VII.

TABLE VII
SVM WEATHER-DATA ONLY METRICS

Class No.	Precision	Recall	F_1 Score	Accuracy
0	0.560	0.7	0.622	0.7
1	0.714	0.75	0.732	0.75
2	0.762	0.8	0.78	0.8
3	0.786	0.55	0.647	0.55
4	0.737	0.7	0.718	0.7
Average	0.712	0.7	0.7	0.7

Comparing table VII to tables V and VI, it can be seen that AlexNet with weather data slightly outperforms the SVM, weather-data only approach. Surprisingly, the weather-data only approach performs approximately the same as the AlexNet without weather data approach.

REFERENCES

[1] F. S. T. J. S.-C. P.-L. Boulent Justine, ”Convolutional Neural Networks for the Automatic Identification of Plant Diseases,” Frontiers in Plant Science, vol. 10, p. 941, 2019.
[2] PyTorch, ”torchvision.models,” [Online]. Available: https://pytorch.org/docs/stable/torchvision/models.html. [Accessed 3 March 2020]
[3] R. Gombru, ”Concatenate layer output with additional input data,” June 2018. [Online]. Available: https://discuss.pytorch.org/t/concatenate-layer-output-with-additional-input-data/20462. [Accessed 3 March 2020].