A CNN-Based Smart Farm Image Analysis System

Hongzhuo Chen, YQJ, and ZNZ

Southeast University, Nanjing, China

Abstract. Convolutional Neural Network (CNN), a kind of artificial neural network with multiple layers, has become popular in machine learning problems, especially the applications that deal with image data over the past decade. In this paper, we adopted CNN to a smart farm image analysis system, which can classify a strawberry as ripe, part-ripe, or unripe, and estimate its acidity and Brix (a value used to quantify its taste) based on its image. During the training process, the training loss (cross-entropy loss) of the classification task converged to 0.03, and the training loss (mean squared error loss, MSE loss) of the estimation task converged to 1.2.

Keywords: Deep Learning, Convolutional Neural Network (CNN)

1 Introduction

Convolutional Neural Network (CNN) has had groundbreaking breakthroughs in various fields, from image processing to voice recognition. One aspect of CNN is to obtain features when input propagates toward the deeper layers. In image classification, the edge might be detected in the first layers, the simpler shapes in the second layers, and then the higher-level features such as faces in the next layers.[1–3]

One of the most well-known applications of CNN is image classification. Yann LeCun et al.[4] designed LeNet used for handwritten and machine-printed character recognition. In our work, we adopted LeNet to classify images of strawberries with different ripening stages (ripe, unripe, or part-ripe). To improve our model's accuracy, we increased our training epochs until training loss, calculated by the cross-entropy loss function, converged to a value (0.03 in our work).

As for estimating the Brix and acid of a strawberry based on its image, we can also apply CNN considering its ability in obtaining abstract features by modifying the size of output samples in the deepest layers of the CNN. We designed an MSE loss function to calculate training loss, and after 5 epochs, the training loss converged to 1.2. Due to the lack of testing data, we utilized K-Fold cross-validation[5] during our training.

2 Background

In this section, we introduce convolutional neural network and it's architecture, and LeNet, a classic and simple CNN. We also introduce Cross Validation, a statistical method we applied in our work due to the lack of testing set.

2.1 Convolutional Neural Network (CNN)

A CNN is a standard NN (neural network) that extends across space via shared weights. CNN recognizes images by having convolutions, which can obtain abstract features, within the input image. Traditional CNN consists of multiple layers, including convolutional layers, pooling layers, fully-connected layers, and non-linearity layers. Fully-connected layers and convolutional layers have parameters while pooling layers and non-linearity layers have not.[6] Fig. 1 shows what features CNN learned.

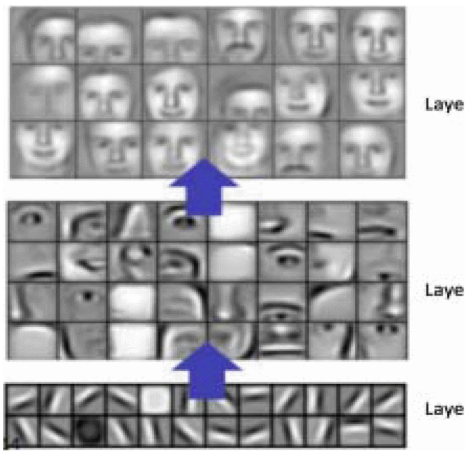

Fig. 1. Learned features from a convolutional neural network

Convolutional layers, the key component of CNN, detect the presence of a set of features in the images received as input. This is achieved by convolution filtering: the principle is to drag a window representing the feature on the image and calculate the convolution product between the feature and each portion of the scanned image. A feature is then seen as a filter: the two terms are equivalent in this context.

In most cases, pooling layers are placed between two convolutional layers. A polling layer receives several feature maps and applies the pooling operation to them to reduce the size of the images while preserving their characteristics.

Fully-connected layers are not characteristic of a CNN since a fully-connected layer is always the last layer of a neural network. This type of layer receives an

input vector and produces a new output vector. It applies a linear combination and then probably an activation function to the input values received. The last fully-connected layer classifies the image as an input to the network: it returns a vector of size N, where N is the number of classes in our image classification problem. Each element indicates the probability for the input image to belong to a class.

In practice, non-linear layers use a ReLU function defined by $ReLU(x) = max(0, x)$. A non-linear layer replaces all negative values received as inputs by zeros and acts as an activation function.[7]

LeNet LeNet, a classic CNN developed by Yann LeCun et al is one of the earliest CNNs and promoted the development of machine learning. Usually, LeNet refers to LeNet-5 proposed in [4]. LeNet consists of 7 layers including input layers. Layer 1 is a convolution layer with six convolution kernels of 5x5, and the size of feature mapping is 28x28. Layer 2 is the subsampling/pooling layer that outputs 6 feature graphs of size 14x14. Layer 3 is a convolution layer with 16 5-5 convolution kernels. The input of the first six Layer 3 feature maps is each continuous subset of the three feature maps in Layer 2, the input of the next six feature maps comes from the input of the four continuous subsets, and the input of the next three feature maps comes from the four discontinuous subsets. Layer 4 is similar to 2, with the size of 2x2 and output of 16 5x5 feature graphs. Layer 5 is a convolution layer with 120 convolution kernels of size 5x5. Each cell is connected to the 5*5 neighborhood on all 16 feature graphs of Layer 4. Here, since the feature graph size of S4 is also 5x5, the output size of Layer 5 is 1*1. So Layer 4 and Layer 5 are completely connected. Layer 6 is fully connected to Layer 5, and 84 feature graphs are output.[4, 8] Fig. 2 shows the architecture of LeNet-5.

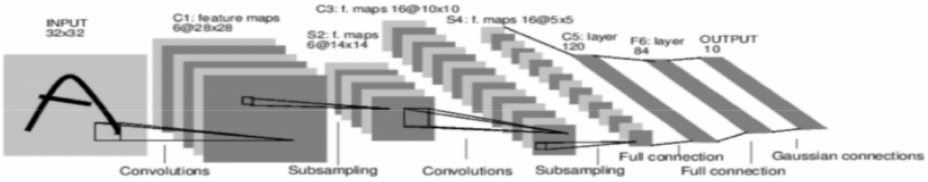

Fig. 2. Architecture of LeNet-5

2.2 Cross-Validation

Cross-validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one for learning or training a model and the other for validating the model. In k-fold cross-validation, the data is first partitioned into k equally (or nearly equally) sized segments or folds.

Subsequently, k iterations of training and validation are performed such that within each iteration a different fold of the data is held out for validation, while the remaining k-1 folds are used for learning.[5] Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.[9] Fig. 3 shows the procedure of 3-fold cross validation.

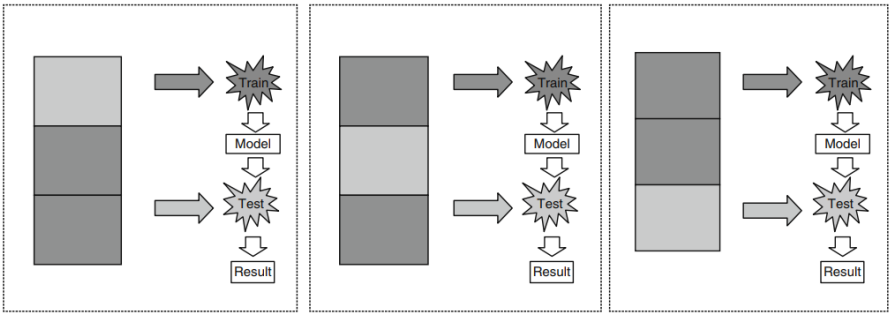

Fig. 3. Procedure of 3-fold cross validation

3 Main Design

In this section, we first describe the overview of the task, then explain our work.

3.1 Smart Farm Image Analysis Task

In this task, 502 images of strawberries are provided with their ripening stage (unripe, ripe, or part-ripe), acidity, and Brix (a value to quantify taste). The task is to develop a pattern recognition model that can classify the images of strawberries as unripe, ripe, or part-ripe, and estimate the Brix and acidity value of any new images of strawberries.

3.2 Smart Farm Image Analysis Model based on CNN

This task can be divided into two subtasks: classification and estimation. Both use some abstract features of the image. Given that CNN can obtain abstract features from the input data, we applied CNN to both the classification and the estimation model.

Classification In the classification task, we applied LeNet with an Adam optimizer[10] whose learning rate is 0.001. As for calculating loss, we used the Cross-Entropy Loss function.[11]

Estimation In the estimation task, we modified a parameter of the last layer, a fully connected layer of our CNN. We set the parameter out_features to 2, which means the CNN output a two-dimensional vector as features. In this way, our model can estimate two values: acidity and the Brix, at one time. Our estimation model uses an Adam optimizer with the learning rate of 10^{-4} and the mean squared error(MSE) function[11] to calculate the loss. If a vector of n predictions is generated from a sample of n data points on all variables, and Y is the vector of observed values of the variable being predicted, with $\hat{Y}$ being the predicted values (e.g. as from a least-squares fit), then the within-sample MSE of the predictor is computed as

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Due to the absence of testing data, we trained our model using 5-fold cross-validation in our task.

4 Performance Evaluation

In this section, we introduce how we evaluate our models.

4.1 Classification

In the classification task, we observe the loss after each training batch. After training 50 epochs, the training loss converged to about 0.03. Fig. 4 shows the training loss of our classification model after each batch of data.

4.2 Estimation

In the estimation task, we calculate the average loss of each fold of training data after each epoch. After training about 5 epochs, the training loss converged to about 1.2. Fig. 5 shows the average training loss of each fold of training data after each epoch. Besides that, we also picked out some images randomly and showed the ground-truth Brix and acidity values compared to the estimated values of our model. Fig. 6 shows some random images of strawberries, GT B & A stands for "ground-truth Brix and acidity", and "Est" means "estimated Brix and acidity of our model"

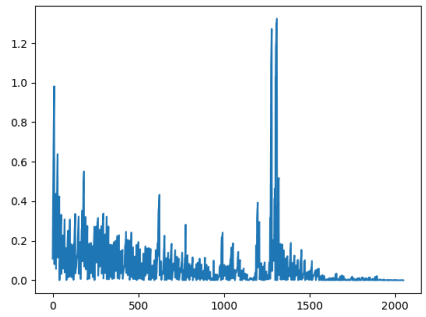

Fig. 4. Training loss after each batch

Fig. 5. Average training loss of each fold of training data after each epoch

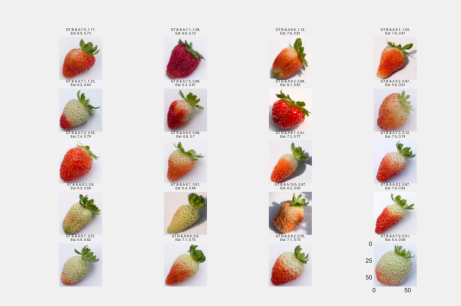

Fig. 6. Some random images of strawberries, GT B & A stands for "ground-truth Brix and acidity", and "Est" means "estimated Brix and acidity of our model"

References

1. Albawi S, Mohammed T A, Al-Zawi S. Understanding of a convolutional neural network[C]//2017 International Conference on Engineering and Technology (ICET). Ieee, 2017: 1-6.
2. Abdel-Hamid O, Deng L, Yu D. Exploring convolutional neural network structures and optimization techniques for speech recognition[C]//Interspeech. 2013, 11: 73-5.
3. Dumoulin V, Visin F. A guide to convolution arithmetic for deep learning[J]. arXiv preprint arXiv:1603.07285, 2016.
4. LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.
5. Refaeilzadeh P, Tang L, Liu H. Cross-validation[J]. Encyclopedia of database systems, 2009, 5: 532-538.
6. Abiodun O I, Jantan A, Omolara A E, et al. State-of-the-art in artificial neural network applications: A survey[J]. Heliyon, 2018, 4(11): e00938.
7. Understand the architecture of CNN, https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7
8. LeNet, https://en.wikipedia.org/wiki/LeNet
9. A Gentle Introduction to k-fold Cross-Validation, https://machinelearningmastery.com/k-fold-cross-validation/
10. Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.
11. Cross entropy, https://en.wikipedia.org/wiki/Cross_entropy
12. Mean squared error, https://en.wikipedia.org/wiki/Mean_squared_error