Convolutional Neural Network Based Landcover Analysis of Satellite Images

William Song Liu

1 Stanford Online High School, Stanford, California, 94305, USA

Abstract. This paper introduces the application of convolutional neural networks (CNNs) for landcover analysis of satellite images. The development of remote sensing techniques makes abundant aerial color and hyperspectral imagery accessible, which brings challenges to efficient data analysis. As a powerful feature representation learning method, CNN is a good candidate for such tasks. Several pre-processing methods and network structures are investigated in this paper. They are able to achieve high accuracy on testing data. The results suggest that network proposed in this paper can be used as an accurate and efficient system to classify landcover usage.

1. Introduction

The first computer algorithm resembling a neural network is the perceptron algorithm, which has been invented in 1957 by Frank Rosenblatt and implemented using the Mark I Perceptron Machine (Rosenblatt 1958). One layer of inputs and one layer of outputs are fully connected with each node in the input having a direct connection to each node in the output. Connections are multiplied by a trainable weight, as well as the output perceptron having trainable biases that determine their activation threshold. The action of optimizing weights and biases through a learning process is the essence of machine learning. During the training process, a perceptron that correctly influences the final prediction has its weight increased, and a perceptron that incorrectly influences the final prediction has its weight decreased. The Mark I Perceptron was designed as a machine to perform basic image recognition, and it could successfully classify simple 20 pixels by 20 pixels images. However, the major limitation to the single-layer perceptron model is that it is not able to recognize non-linear patterns. Most notably, Marvin Minsky and Seymour Papert (Minsky and Papert 1969) pointed out that a single-layer perceptron model was unable to learn the XOR function, while a multilayer perceptron model was able to.

Modern-day neural networks, with the introduction of hidden layers between the input layer and the output layer, are able to extract increasingly abstract features from the previous layer of data. Increasing the depth of a neural network is proven to be extremely useful in the cases of AlexNet (Krizhevsky et al. 2012) and VGGNet (Simonyan and Zisserman 2015). However, the usage of multiple layers creates a new problem of optimizing the weights of each of the connections. While every perceptron's influence on the outputs could directly be determined in a single-layer perceptron model, the optimization of each outgoing connection in a multilayer model would depend on the previous layer's incoming connections as well. This leads to the invention of backpropagation using the chain rule in Calculus, which is used to calculate the derivative of every neuron that feeds into a given neural, and thus, the error can be propagated to each of the neurons (Rumelhart et al. 1988). Neural networks generally perform better when they have more hidden layers.
Even though normal neural networks are theoretically capable of solving complex problems like image recognition, there is a more practical and efficient algorithm specifically designed to do image recognition, the convolutional neural network or CNN. The first section of the convolutional neural network is composed mainly of convolutional layers. Convolutional layers examine the previous layer's output image one section at a time, which is described as using filters with a given size convolving with a given stride. This acts as if the neural network is trained on many small separate images and each focusing on a different part of the input image. Since each layer convolves over the previous output, the neural network does not have to learn the same features independently for every section of the image. This is the main advantage of convolutional neural networks over normal neural networks: by ignoring spatial or positional differences in the data, the neural network is able to use fewer parameters. In addition, the cascading of convolutional layers allows the learning of hierarchical features. After several convolutional layers, there is usually a pooling layer. Pooling is used to reduce the size of the data while maintaining the most important features. For example, max pooling saves the maximum number in each moving filter and discards the rest of the data. The last few layers of the convolutional neural network are fully connected layers. This is used to translate the previous results of the neural network into a classification output. Because they are connected with all neurons of the previous layer, they are able to make decisions based on global information, unlike the convolutional layers which only look at sections of local information. Assessing global information for the final classification is necessary to have a holistic view of the input image.

There have been many recent developments in image recognition tasks and convolutional neural networks. The MNIST database of handwritten digits is a starting point in the field of image recognition, which Wan Zhu uses to demonstrate the better performance of convolutional neural networks compared to fully connected backpropagating neural networks (Zhu). There have been more recently developed large datasets of labelled images, which are necessary for the training and evaluation of image recognition models. These are presented as challenges, the most notable of which include the PASCAL Visual Object Classes (VOC) Challenge (Everingham et al. 2009) and the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al. 2014). More recent developments in convolutional neural networks include AlexNet, VGGNet, GoogLeNet, and ResNet. AlexNet (Krizhevsky et al. 2012) participated in the 2012 ILSVRC and won first place with a 15.4% top 5 error rate (how often the correct classification appears in the top 5 predictions), beating second place's standard neural network that achieved a 26.2% top 5 error rate. AlexNet was the first convolutional neural network to achieve very good results, popularizing it within the computer vision field. Even though VGGNet (Simonyan and Zisserman 2015) won second place in the 2014 ILSVRC, it demonstrated two important factors of convolutional neural networks: the importance of depth and simplicity. Depth refers to the number of layers in the network, especially convolutional layers in this case. Simplicity refers to the reduced size of filters used in convolutional layers. Even though the filters of VGGNet are significantly smaller than other common models, three layers of 3x3 filters have an effective receptive field of 7x7, so it is able to extract similarly abstract features as other models. Smaller filters have fewer parameters, so more computational resources can be put into using more layers. GoogLeNet (Szegedy et al. 2015), the winner of the 2014 ILSVRC, used a very different, breadth-focused approach. GoogLeNet is composed of modules known as inception modules, which performs convolutions of different filter sizes on the same input and then concatenates the results. Each module expands the network width-wise instead of depth-wise. This solves the problem of not being able to choose an appropriate filter size, as there could be a large variance in the size of objects in different images. However, performing so many convolution operations makes the network computationally costly, so 1x1 convolutions are applied before the 3x3 and 5x5 convolutions to reduce the dimensions of the data. ResNet (He et al. 2016) won first place in the 2015 ILSVRC. It solved the problem of gradients becoming increasingly smaller during the backpropagation in deep networks, by adding the residual from a shallower layer to a deeper layer. The identities of previous layers are carried over in this way for better optimization.
Convolutional neural networks are well suited for the task of image classification in the field of remote sensing. As Zhu et al. describe, “atmospheric scattering conditions, complicated light scattering mechanisms, inter-class similarity, and intra-class variability result in the hyperspectral imaging procedure being inherently nonlinear,” and convolutional neural network is a better choice than conventional machine learning methods in regards to nonlinear image classification (Zhu et al. 2017). However, CNNs also has its own problems when applied to remote sensing. Nogueira et al. compares different cases of applying pre-existing CNNs to new sets of remote sensing data. As it is often difficult or impractical to obtain a large set of labelled images, the model performs poorly and tends to overfit (Nogueira et al. 2017). SAT-6 airborne dataset with 324000 labelled training images and 81000 labelled testing images is used in this paper. Data augmentation using rotations and flips is used to increase the total number of training images to further avoid overfitting problem.

2. Network Design

2.1. Features

The network is composed of three types of layers: convolutional layers, batch normalization layers, and max pooling layers.

Convolutional layers. Convolutional layers are the essence of convolutional neural networks. Each convolutional layer is composed of a set of filters, which convolves across the input image and learns features.

Batch normalization layers. Normalization is a technique that modifies all values in a set of data, such that the distribution of the set will be closer together after normalization. More normalized or less dispersed data speeds up the training process of neural networks, as the weights and biases do not need to be modified by a large amount to fit the data. Another advantage of batch normalization in deep neural networks is that it will make each layer more independent from other layers. When batch normalization is not applied, any change in the weights or biases in one of the shallower layers will result in a change in the output of a deeper layer, thus making the network having to retrain deeper layers based on changes in shallower layers. Batch normalization layers reduce this inefficiency by reducing the amount that the deeper layers have to change.

Max pooling layers. One of the purposes of a pooling layer is to reduce the scale or amount of data, by discarding the unimportant bits and retaining the important bits. Max pooling uses a moving window similar to that in convolutional layers and takes the maximum values in the window of data, thus reducing the output size. Another benefit of pooling layers is decreasing the influence of small translations in the image, as the maximum value of most sections should remain the same.

2.2. Structure

The network structure is based upon the structure of VGGNet (Simonyan and Zisserman 2015), with certain modifications. The structure can be seen in Table 1.

| conv3-24 | conv3-48 | conv3-96 | conv3-192 | conv3-192 | conv3-192 | conv3-192 | conv3-192 | conv3-192 | FC-6 | soft-max |
|---------|----------|----------|-----------|-----------|-----------|-----------|-----------|-----------|------|----------|
| batch norm | batch norm | batch norm | batch norm | batch norm | batch norm | batch norm | batch norm | batch norm |      |          |
| ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU |      |          |
| maxpool | maxpool | maxpool | maxpool | maxpool | maxpool | maxpool | maxpool | maxpool |      |          |

The first discrepancy between this network and VGGNet is the reduced number of filters. As VGGNet was trained on the ImageNet dataset (Deng et al. 2009) with several million images, while this model is trained on the SAT-6 dataset with 324,000 training images, the number of filters in convolutional layers result in overfitting.
The second discrepancy is only having one fully connected output layer with 6 neurons at the end, instead of VGGNet’s 2 layers of 4096 neurons and the output layer with 1000 neurons. As the images in SAT-6 have 4 channels while ImageNet only has 3, using the same structure of the fully connected layers would result in a much larger network. This design was to reduce the network size and also to reduce overfitting.

This design uses a total of 132,480 parameters, with 129,882 of the parameters in convolutional layers and only 2,598 parameters for the fully connected layer. In contrast to VGGNet’s parameter distribution with 89.36% of all parameters belonging to the fully connected layers, this model has significantly reduced the proportion of parameters in the fully connected layer to 1.96%. This is a significant advantage of this model to increase training speed and reduce memory usage.

3. Dataset

The dataset used is SAT-6 airborne dataset, which is extracted from the National Agriculture Imagery Program (NAIP). The extracted data are 28x28 pixels images, with the 4 channels: red, green, blue, and Near Infrared (NIR). Each image is hand-labelled with one of 6 possible labels: building, barren land, trees, grassland, road, or water. There is a total of 324,000 training images and 100,000 testing images in the dataset. Sample images are shown in Figure 1.

![Sample images in the dataset. Six samples per class are shown in each row. The name of the corresponding class is displayed at left.](image-url)
4. Experiments

4.1. Optimizer

Two different optimizers are tested in this paper, the Gradient Descent Optimizer and the Adam Optimizer. The Gradient Descent Optimizer is configured using a learning rate of 0.002, and the Adam Optimizer is also configured using a starting learning rate of 0.002. Table 2 shows the training and testing accuracies of the two models, both based upon the structure described above in 3.2. Structure, or equivalently Model-24 in 5.1. Convolutional filters. The Adam Optimizer reaches a lower minimum point using the decaying learning rate, resulting in higher accuracies. This is consistent with the results presented in Diederik Kingma and Jimmy Ba’s paper, demonstrating that Adam is indeed more effective and efficient for this model with a large amount of data and large number of parameters (Kingma and Ba 2017).

Table 2. Accuracy comparison of models using different optimizers.

| Model name    | Optimizer used | Training accuracy | Testing accuracy |
|---------------|----------------|-------------------|------------------|
| Model-GradDesc| Gradient Descent | 100.00%            | 88.90%           |
| Model-Adam    | Adam           | 100.00%            | 91.32%           |

4.2. Convolutional filters

The number of filters in each convolutional layer makes a significant difference in the testing accuracy of the resulting model. All models have been optimized using Adam Optimizer. See Table 3 for the comparison of the final performance of models.

Table 3. Accuracy comparison of models with different numbers of filters.

| Model name | Number of filters in each convolutional section | Training accuracy | Testing accuracy |
|------------|-----------------------------------------------|-------------------|------------------|
| Model-4    | 4, 8, 16, 32, 32                               | 100.00%           | 80.44%           |
| Model-8    | 8, 16, 32, 64, 64                              | 100.00%           | 85.34%           |
| Model-16   | 16, 32, 64, 128, 128                           | 100.00%           | 91.04%           |
| Model-24   | 24, 48, 96, 192, 192                           | 100.00%           | 91.32%           |
| Model-32   | 32, 64, 128, 256, 256                          | 100.00%           | 90.54%           |

As there is only a minor testing accuracy improvement from Model-16 to Model-24, a further addition of more filters does not seem to improve the testing accuracy. More layers also cause model to overfit. Thus, Model-24 is used as a Baseline model for all future experiments.

4.3. Data Augmentation

The number of instances of each class of data in the dataset were not evenly distributed. The distribution can be seen in Figure 2 in blue. There is a significantly less number of Road and Building samples in comparison with other classes, which resulted in a model that would be significantly biased towards choosing other classes.
To solve this problem, data augmentation was implemented on all images of the Building and Road classes to raise the effective number of Building and Road samples to a level similar to that of Barren Land, Trees, and Grassland. For example, each Building sample is rotated by 90 degrees for 3 times to quadruple the effective amount of Building samples from 14,923 to 59,692. In addition to applying three 90 degree rotations on each Road sample, a horizontal flip and a vertical flip was applied to sextuple the effective amount of Road samples from 8,192 to 49,152. The data distribution after data augmentation is shown in Figure 2 in orange.

The results of apply data augmentation is shown in section 5.1. Confusion matrices.

5. Results

5.1. Confusion Matrices
The confusion matrix of the Baseline model is compared with the confusion matrix of the Data Augmentation model, shown in Figure 3. The Data Augmentation model does improve the accuracies of the Building and Road classes from 47% to 67% and from 34% to 54% respectively. The false positives of Road samples being classified as Building samples decreased from 37% to 28%, most likely because the number of Road samples were increased by a greater factor than the number of Building samples. This method of data augmentation comes at a sacrifice to the accuracy of Grassland, decreasing it from 90% to 73%. Water is relatively easy to be correctly classified, both models yield perfect performance on the Water class.
In Table 4, I computed the average and weighted testing accuracies of the confusion matrices. The average testing accuracy is the sample-wise accuracy. The weighted testing accuracy is computed using the number of classes in the testing dataset as weights. These two metrics are used to compare the model’s performance in average as well as on targets of interest. The average testing accuracy, which is unweighted to the number of classes in the dataset, has a decrease of 3.11% using the Data Augmentation model. The weighted testing accuracy decreased 2.46% using the Data Augmentation model. This shows that the Data Augmentation model does not perform better than the Baseline model when using this specific dataset, nor in other distributions of classes. Thus, data augmentation is not used in the final model.

Table 4. Accuracy comparison of models with different numbers of filters.

| Model Name       | Average Testing Accuracy | Weighted Testing Accuracy |
|------------------|--------------------------|---------------------------|
| Baseline         | 91.32%                   | 73.98%                    |
| Data Augmentation| 88.21%                   | 71.52%                    |

5.2. Accuracies

Table 5 and Figure 4 show the resulting training and testing accuracies of the 4 models been tested with, except the models from testing the number of filters. The training accuracies of all the models converged at 100%, but the testing accuracies show no sign of overfitting. As the model with the highest final testing accuracy is Batch Norm, it is the final model presented.

Table 5. Training and testing accuracies comparison of all models from the tests ranked by descending testing accuracy, except the number of filters test.

| Model name   | Training accuracy | Testing accuracy |
|--------------|-------------------|------------------|
| Batch Norm   | 100.00%           | 92.01%           |
| Baseline     | 100.00%           | 91.32%           |
| Grad Desc    | 100.00%           | 88.90%           |
| Data Aug     | 100.00%           | 88.21%           |

Figure 4. Training and testing accuracies of the models in Table 1 in visual form.
6. Conclusion
In this work, a convolutional neural network is trained to classify landcover usage by using satellite aerial color and hyperspectral images. Throughout this process, I discovered an appropriate set of hyperparameters and experimented with various features, attempting to further improve model performance. The experiments conducted were mostly using a trial and error method, to test variations of a Baseline model and preserve modifications that improved the performance of the model. The first test I conducted was to determine which type of optimizer would yield the best result, and I concluded that the Adam Optimizer performs better than the Gradient Descent Optimizer in all factors, including the final testing accuracy. The next experiment conducted was for the number of filters in each convolutional layer. While all models were able to achieve a 100% training accuracy, the models with more filters resulted in a higher testing accuracy, with the testing accuracy peaking at Model-24. Model-36 resulted in over-fitting and achieved a lower testing accuracy. Thus, Model-24 is used as the Baseline model for all other experiments. Next, I tried to apply data augmentation by upscaling the number of Building and Road samples to make the distribution of classes more balanced. This came at a cost of sacrificing accuracy of the Grassland class, and it did not end up yielding an improved average or weighted accuracy, so I did not include it in our final model. Finally, I applied batch normalization on the Baseline model, which yielded the highest accuracy of all models, at 92.01%.

References
[1] F. Rosenblatt, “Principles of neurodynamics. perceptrons and the theory of brain mechanisms,” 1961.
[2] Minsky, M., & Papert, S. (1969). Perceptrons: An introduction to computational geometry.
[3] A. Krizhevsky, “Learning Multiple Layers of Features from Tiny Images,” Sci. Dep. Univ. Toronto, Tech. pp. 1–60, 2009.
[4] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” pp. 1–14, 2014.
[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, no. 6088, p. 533, 1986.
[6] Zhu, W. (2018) Classification of MNIST Handwritten Digit Database using Neural Network. http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_117.pdf
[7] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. International journal of computer vision, 88(2), 303–338.
[8] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” 2009 IEEE Conf. Comput. Vis. Pattern Recognit., pp. 248–255, 2009.
[9] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D. & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).
[10] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[11] Zhu, X. X., Tuia, D., Mou, L., Xia, G. S., Zhang, L., Xu, F., & Fraundorfer, F. (2017). Deep learning in remote sensing: A comprehensive review and list of resources. IEEE Geoscience and Remote Sensing Magazine, 5(4), 8-36.
[12] Nogueira, K., Penatti, O. A., & dos Santos, J. A. (2017). Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recognition, 61, 539-556.
[13] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” pp. 1–15, 2014.