Machine Learning is Fun Part 3

Yuan An

In the last article, it is mainly about the neural network. The third part of *Machine Learning is Fun* [1] is talking about recognizing objects with Deep Learning.

A little kid can figure out the bird from other items, but it is difficult for the computer to do this. That makes the very best computer scientists puzzled for over 50 years.

In the last few years, a good approach to object recognition using deep convolution neural networks has been founded, and there are more information about this in the paper of Krizhevsky *et al*. [2].

Then the author began object recognition from an easy example – recognizing the handwritten number 8. In the part 2, there is a small neural network to estimate the price of a house based on how many bedrooms it had, how big it was, and which neighborhood it was in. And we know that the idea of machine learning is that the same generic algorithms can be reused with different data to solve different problems. So this neural netork (see Fig. 1) can be modified to recognize handwritten text.

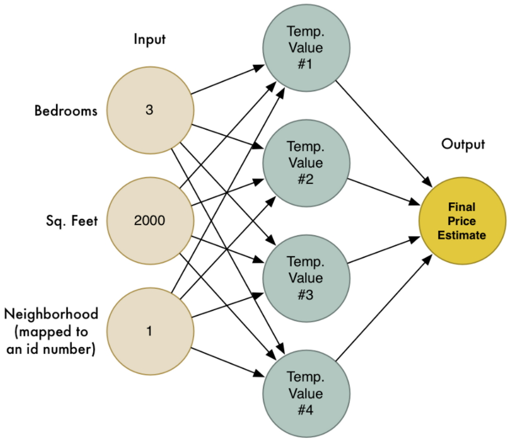

Figure 1. The model of predicting house sale price

Machine learning only works when data is provided. In the article, the author uses MNIST dataset which provides 60,000 images of handwritten digits, each as an 18*18 pixels. There are some 8s from the data set (see in Fig. 2)

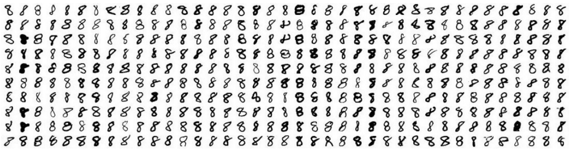

Figure 2. Some 8s from the MNIST data set

Now we have data, and how do we put the data in the algorithm. The author told us that everything is just numbers. A neural network takes numbers as input. To the computer, an image is really just a grid of numbers that represent how dark each pixel is. Because each image has 18*18 pixels, so that is an array of 324 dimension numbers. To handle those inputs, the neural network should be enlarged to have 324 input nodes.

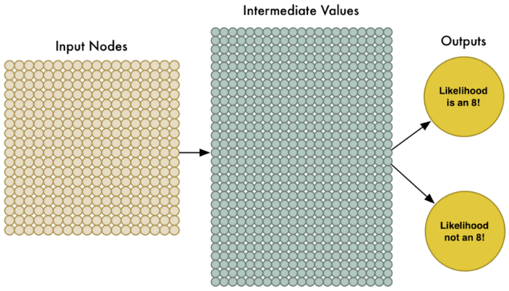

Figure 3. Modified neural network

As shown in Fig. 3, the neural network has two outputs now that the image is an “8” and the image is not “8”. By having a separate output for each type of object to be recognized. So we can use a neural network to classify objects into groups. And now all that is left is to train the neural network with image of 8s and not-8s so it learns to tell them apart.

Figure 4. Training data

Here are some of training data in Fig. 4. When training the neural network, we should tell it the probability the image is an “8” is 100 % and the probability it is not an “8” is 0 %. Vice versa for the counter-example images.

The good news is that the recognizer does work well on

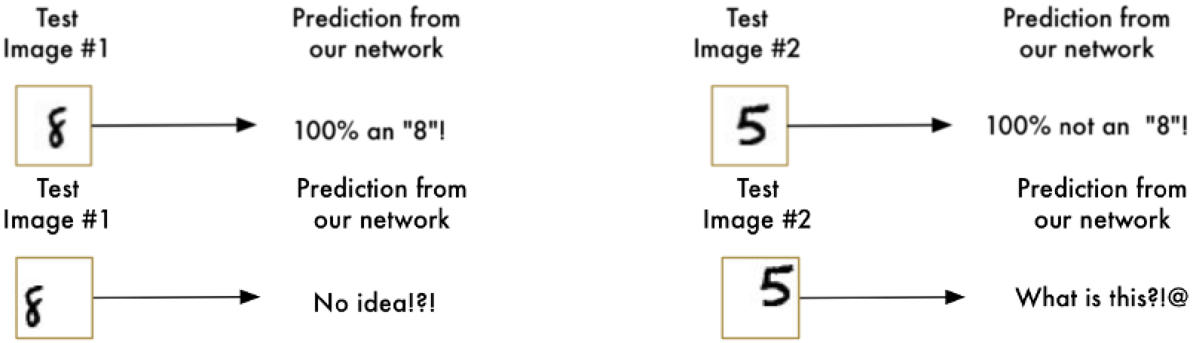

Figure 5. Good news and bad news

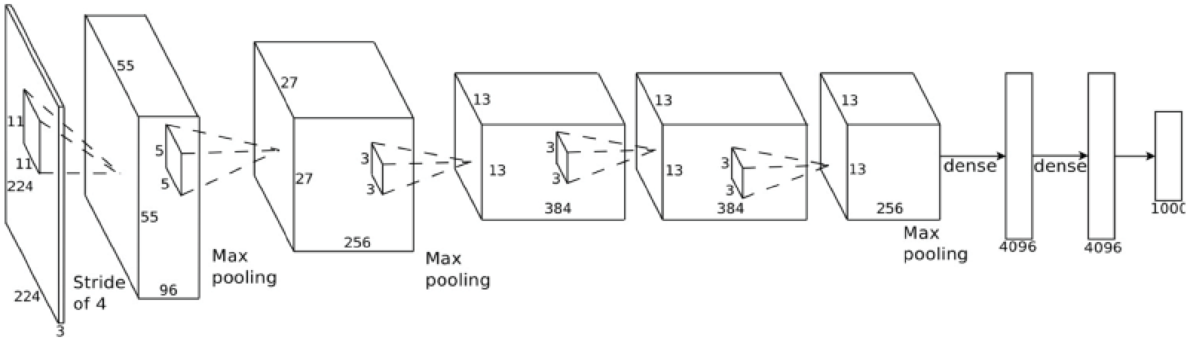

Figure 6. A realistic deep convolution network.

simple images where the letter is right in the middle of the image, but the bad news is that the recognizer totally fails to work when the letter isn't perfectly centered in the image.(see Fig. 5). To solve the problem, the author proposed two solutions. One is searching with a sliding window, the other is using more data and training a deep neural network. But those two methods are all inefficient.

There is a good way to make the neural network be smart enough to know that an "8" anywhere in the picture is the same thing without all that extra training.

The solution is convolution. As a human, we intuitively know that pictures have a hierarchy or conceptual structure. We can instantly recognize the hierarchy in one picture. *E.g*. there is a picture of environmental portrait, we can figure out sightseeing and people in the picture. Most importantly, we recognize the idea of people no matter what the circumstance the people is on. And we don't have to re-learn the idea of people for every possible circumstance it could appear on.

But now, the neural network we trained can't do this. It thinks that an "8" in a different part of the image is an entirely different thing. So a concept of translation invariance should be introduced to make the net know. We can using a process called convolution to approach the goal.

Convolution work in the following steps:

1. Break the image into overlapping image tiles.
2. Feed each image tile into a small neural network.
3. Save the results from each tile into a new array.
4. Downsampling.
5. Make a prediction.

Then add more steps: convolution, maxpooling, and finally a fully-connected network. Here's what a more realistic deep convolution network looks like Fig. 6.

References

[1] A. Geitgey. Machine learning is fun. `https://medium.com/@ageitgey/machine-learning-is-fun-80ea3ec3c471`.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.