Example 78 (cont'd): Examples of Misses

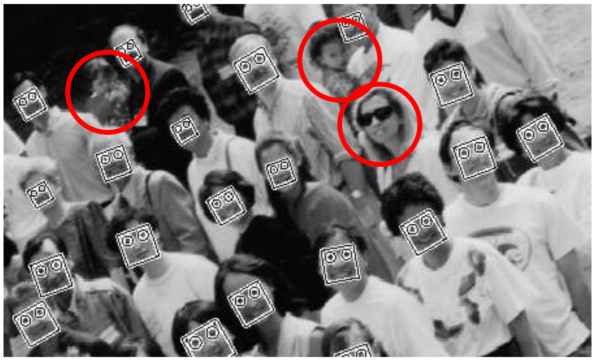


Figure from H. A. Rowley, S. Baluja and T. Kanade, "Rotation Invariant Neural Network-Based Face Detection," Proc. CVPR, ©1998, IEEE

Neural networks can be effective when we impose additional architectural constraints on their design. We now consider an example of what has come to be called a "convolutional neural network."

Convolutional Neural Networks

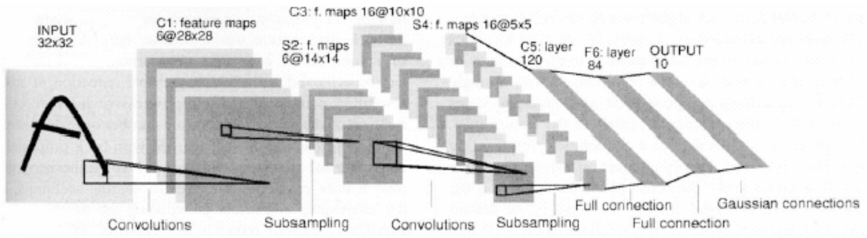
- Template matching using NN classifiers seems to work
- Natural features are filter outputs

— probably, spots and bars, as in texture

— but why not learn the filter kernels, too?

As an example, consider the convolutional neural network, LeNet. As stated in their 1998 paper, Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, ©1998, IEEE, LeNet "is in commercial use in the NCR Corporation line of check recognition systems for the banking industry. It is reading millions of checks per month in several banks across the United States."

Example 79: Recognizing Handwritten Characters



Forsyth & Ponce Figure 22.17

Figure from Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, ©1998, IEEE

LeNet 5 has seven layers, not counting the input, all of which contain trainable parameters (i.e., weights). The layers filter, subsample, filter, subsample, and finally classify based on outputs of this process. The layers marked "C" are convolutional layers; those marked "S" are subsampling layers and the one marked "F" is fully connected. The general form of the classifier uses an increasing number of features at increasingly coarse scales to represent the image window. Finally, the window is passed to a fully connected neural net, which produces a rectified output that is classified by looking at its distance from a set of canonical templates for characters.

The input plane receives images of characters that are approximately size normalized and centered.

Units in the first hidden layer are organized in six planes, each of which is a feature map. A unit in a feature map has 25 inputs connected to a 5×5 area in the input, called the receptive field of the unit. Each unit has 25 inputs and therefore 25 trainable coefficients plus a trainable bias. The receptive fields of contiguous units in a feature map are centered on corresponding contiguous units in the previous layer. Therefore, receptive fields of neighboring units overlap.

The second hidden layer is a subsampling layer. This layer comprises six feature maps, one for each feature map in the previous layer. The receptive field of each unit is a 2×2 area in the previous layer's corresponding feature map. Each unit computes the average of its four inputs, multiplies it by a trainable coefficient, adds a trainable bias, and passes the result through a sigmoid function.

Successive layers of convolutions and subsampling are typically alternated resulting in a "bipyramid." At each layer, the number of feature maps is increased as the spatial resolution is decreased. Each unit in the third hidden layer may have input connections from several feature maps in the previous layer.

Example 79 (cont'd):

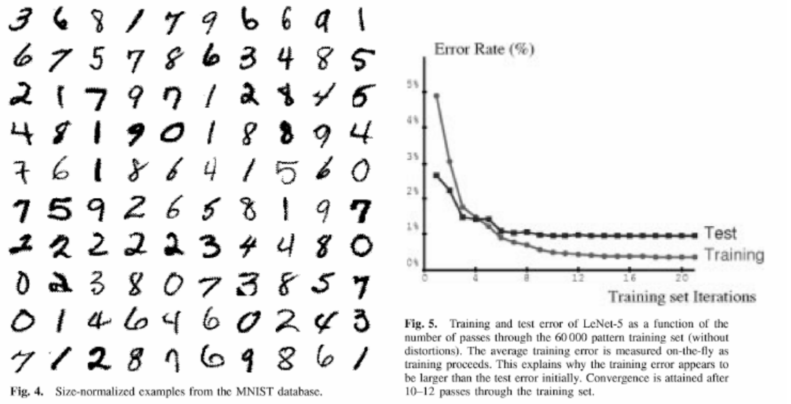


Fig. 4. Size-normalized examples from the MNIST database.

Fig. 5. Training and test error of LeNet-5 as a function of the number of passes through the 60 000 pattern training set (without distortions). The average training error is measured on-the-fly as training proceeds. This explains why the training error appears to be larger than the test error initially. Convergence is attained after 10–12 passes through the training set.

Forsyth & Ponce Figure 22.18

Figures from Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, ©1998, IEEE

The sub-figure on the left shows a small subset of the MNIST database of handwritten characters, used to train and test LeNet 5. There is wide variation in the appearance of each character. The sub-figure on the right shows the error rate of LeNet 5 on a training set and on a test set, plotted as a function of the number of gradient descent passes through the entire training set of 60,000 examples (i.e., if the horizontal axis reads six, the training has taken 360,000 gradient descent steps). Note that at some point the training error goes down but the test error doesn't; this happens because the system's performance is optimised on the training data. A substantial difference would indicate "overfitting."

Support Vector Machines