Gradient-based Learning Applied to Document Recognition

Yann LeCun, Leon Bottou, Yoshua Bengio and Patrick Haffner

Presenter: Lu Jiang
Outline

• Introduction
• Convolutional Neural Network
• Multiple Characters Recognition
• Conclusions
Outline

• Introduction
• Convolutional Neural Network
• Multiple Characters Recognition
• Conclusions
Introduction

• Key message: better pattern recognition systems can be built by relying more on automatic learning and less on hand-designed heuristics.
  – Hand-crafted features vs. learned features

Why learned features:
• (a) lower dimension can be easily compared
• (b) invariant to transformations and distortions of the input patterns.
Background

• LeCun’s most cited paper.
• This paper published in 1998, at that time
  – SVM (and kernel learning) are quite popular.
  – Hand-crafted features (e.g. SIFT) are dominant.
  – MNIST (58k images) is a big and challenging data.
• This paper did not become popular until 2012, when the proposed convolutional neural networks were successfully applied on ImageNet challenge (AlexNet).
• Now almost every deep learning network for visual recognition uses convolutional layers.
MNIST
(NIST handwritten digit database)
Outline

• Introduction
• **Convolutional Neural Network**
• Multiple Characters Recognition
• Conclusions
Intuitions

• A network can be fed with the pixels in raw images (fully-connected layer) but:
  – Formidable number of parameters. $256 \times 256 \times 3 = 196K$ parameters. **Overfitting!**
  – Sensitive to size, shift slant, position variations caused by, or example, resized images.
  – Topology in images are ignored. The fact is local nearby pixels are highly correlated. Local points can form edges, end-points, and corners.

• **Proposed approaches**: learn pattern that can be positioned at various locations; force learned pattern are from local pixels.

• **Solution**: Convolutional Neural Network
  – Convolution layer and pooling layer are inspired by “simple” and “complex” cells [Fukushima et al 1982].
Convolutional Neural Network

- Input: inputs from a set of units located in a small neighborhood in the previous layer. First conv layer receives the resized and normalized images.
- Output: a number of feature maps (holding neurons arranged in a 3D volume)
Convolutional Neural Network

- Input: inputs from a set of units located in a small neighborhood in the previous layer. First conv layer receives the resized and normalized images.
- Output: a number of feature maps (holding neurons arranged in a 3D volume)
Convolutional Neural Network

• Input: inputs from a set of units located in a small neighborhood in the previous layer. First conv layer receives the resized and normalized images.

• Output: a number of feature maps (holding neurons arranged in a 3D volume)

The learnable parameter in the same feature map is shared. (weight sharing)
To learn meaningful pattern at different locations.
Convolutional Neural Network

• Input: inputs from a set of units located in a small neighborhood in the previous layer. First conv layer receives the resized and normalized images.

• Output: a number of feature maps (holding neurons arranged in a 3D volume)

pooling (sub-sampling) layer

Down-sampling the input and preserve meaningful statistics (average or max pooling). Make learned pattern more invariant.
Convolutional Neural Network

- Input: inputs from a set of units located in a small neighborhood in the previous layer. First conv layer receives the resized and normalized images.
- Output: a number of feature maps (holding neurons arranged in a 3D volume)
Convolutional Layer in 1D

\[ z_1 = w_1 x_1 + w_2 x_2 + b \]
Convolutional Layer in 1D

\[ z_2 = w_1 x_2 + w_2 x_3 + b \]
Convolutional Layer in 1D

\[ z_3 = w_1 x_3 + w_2 x_4 + b \]

\( \approx \) measures the dot product similarity to the local inputs.

Our goal is to learn the parameters \( w \)

The hope is we can learn patterns frequently occurred in the inputs.
Convolutional Layer in 1D

\[ z_3 = w_1 x_3 + w_2 x_4 + b \]
Convolutional Layer in 1D

Smaller strides work better in practice [Fei-fei Li et al. 2015].
Convolutional Layer in 2D

Input filter weights feature map

Forward path

Fei-fei Li et al. 2015
Convolutional Layer in 2D

Input

Filter weights

Feature map

Fei-fei Li et al. 2015
Convolutional Layer in 2D

The learnable parameter in the same feature map is shared. (weight sharing)
To learn meaningful pattern at different locations.

Fei-fei Li et al. 2015
Pooling (Sub-sampling) Layer

In this paper, Lecun used a linear transformation. Weighted sum of the inputs plus a bias term. Max pooling becomes quite popular nowadays.

Down-sampling the input and preserve meaningful statistics (average or max pooling). Make learned pattern more invariant.
• C1 layer has 6 feature maps (28x28), a 5x5 receptive field resulting in $(5*5+1)*6 = 156$ learnable parameters which are from $28*28*(5*5+1)*6 = 122,304$ connections.
Convolutional Neural Network

- S1 layer has 6 feature maps (14x14), a 2x2 receptive filed resulting in $(1+1)\times6 = 12$ learnable parameters which are from $14 \times 14 \times (2 \times 2 + 1) \times 6 = 5,880$ connections.

Fewer parameters but computationally intensive to compute (#connections result from convolution)
Convolutional Neural Network Training

• Compute partial derivatives of the loss function with respect to each connection, as if there were no weight sharing. [via backprop]

• Aggregate the partial derivatives of all connections that share a same parameter. Update the parameter with the aggregated derivatives.
LeNet-5 to AlexNet
Results on MNIST

- Linear: 12.0
- Pairwise: 7.6
- SVM: 3.3
- Convolutional Neural Net: 4.5
- Neural Net: 4.7
Results on MNIST

2-3 days to train 20 iterations.
Multi-Module Recognition System

• Object-Oriented Design:
  – Every module can be a layer. For example, loss function layers, convolutional layers, and even graph transformers.
  – Every layer is an object that has functions like fprop and bprop.
  – Complex systems can be built upon those simple layers, and trained by gradient-based learning algorithms.

• Inspire the design of deep learning tools like caffe, Torch
Outline

- Introduction
- Convolutional Neural Network
- Multiple Characters Recognition
- Conclusions
• Over-segmentation: generate a large number of different (probably incorrect) segments.
• Segmentation graph: an arc between two nodes indicate a segment result
• A complete path between start and end node contains each piece of ink once and only once
Recognizing multiple characters
Graph Transformation Networks

A path indicates a possible interpretation of the input word

Find the optimal path

Calculate loss

LeNet-5
Recognizing multiple characters: Graph Transformation Networks

• To train GTN:
  – Add a layer called a path selector to select the paths with **correct label sequence** in the interpretation graph.
  – Calculate the penalty.
  – Back propagate the penalty to the neural network.
  – How to calculate the partial gradient with respect to graphs?
    • Define a binary function. Assign gradient 0 for the arcs not in the correct/optimal path. 1 otherwise.
Recognizing multiple characters: Displacement Neural Network

Interestingly, LeCun mentioned RNN but did not use it because he said it is hard to train.
Outline

• Introduction
• Convolutional Neural Network
• Multiple Characters Recognition
• Conclusions
Conclusions

• The most representative work of LeCun. A seminal on neural networks for visual recognition.
• This paper proposed several interesting notations:
  – Hand-crafted features should be replaced by learned features.
  – Large-sized systems can be learned by gradient-based method with efficient back propagation.
  – Proposed the notation of graph transformer layer that can be plugged into a network.
Thank you.
Any Questions?