Neocognitron

Fukushima, K. and Miyake, S. 1981

presented by Sam Thomson
Oct. 3, 2013
Overview

• multilayer neural network inspired by the mammalian visual system

• unsupervised image classification, tolerant to shifts and deformations

• improvement on the cognitron

Kunihiko Fukushima
Task

• Unsupervised handwritten character recognition
  • input - unlabeled images
  • output - vector, with each bit hopefully encoding a distinct class of images
Design - High Level

- multiple (usually about 3 hidden) layers
- Successive layers recognize higher-level patterns
Design - Makeup of a Layer

- each layer has $k$ $S$-planes
- each $S$-plane feeds into its own $C$-plane
- $V_S$-planes and $V_C$-planes inhibit $S$-planes and $C$-planes, respectively
Design - S-plane

- cells in each plane are arranged in a 2-d grid
- each S-cell looks at a sliding 2-d window in the previous layer
- S-cells in a plane all have the same coefficients (i.e. they are convoluted), but look at a different window
Design - S-plane
Design - S-plane
Design - C-plane

- an S-plane learns to recognize one feature no matter where it is
- the corresponding C-plane ORs a region of S-cells to recognize that feature anywhere in that region (achieving a level of shift invariance)
- C-cell input weights are not learned
Design - C-plane
Design - C-plane
Design - Output

- In the final layer, each C-plane has only one cell, which effectively looks at the entire image
Design - All Together Now
Learning
Learning - Cognitron

- weights get initialized with small positive values
- for each training instance, if a cell is the most active in its region and in its plane, then its active weights get reinforced
- show the same few training instances over and over again
Learning - Cognitron
Learning - Cognitron

- similar to Hebbian learning ("fire together, wire together"), but only one cell maximum per layer and region gets reinforced

- note: we’re not doing gradient descent, and not minimizing any objective
Learning

- mostly glossing over inhibitor cells and mathematical formulas. refer to paper
- math works out so that an S-cell’s weights directly correspond to the feature it is recognizing, and activation = cosine similarity
Fig 12 Receptive fields of the cells of each of the 24 S-planes of layer $U_{S1}$, which has finished learning.
Learning - Example Activations

Fig 10  Response of the cells of layers $U_0$, $U_{C1}$, $U_{C2}$ and $U_{C3}$ to each of the five stimulus patterns
Discussion
Discussion

- Experiment was a toy problem. Does this work on anything real?
- Does it need to be so complicated?
Problems - Not Really Scale-Invariant

• the amount of shift/deformation-invariance is hardcoded into the structure, by how big a region each C-cell covers e.g.

• intuitively: only one training example is used for each digit; how could it possibly be learning what kinds of deformations to allow?

• empirically demonstrated by Barnard and Casasent, 1990
References


Questions?