CSC321: 2011
Introduction to Neural Networks and Machine Learning

Lecture 6: Applying backpropagation to shape recognition

Geoffrey Hinton

Applying backpropagation to shape recognition

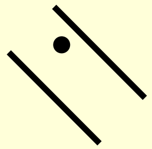
- People are very good at recognizing shapes
  - It's intrinsically difficult and computers are bad at it
- Some reasons why it is difficult:
  - Segmentation: Real scenes are cluttered.
  - Invariances: We are very good at ignoring all sorts of variations that do not affect the shape.
  - Deformations: Natural shape classes allow variations (faces, letters, chairs).
  - A huge amount of computation is required.

The invariance problem

- Our perceptual systems are very good at dealing with invariances
  - translation, rotation, scaling
  - deformation, contrast, lighting, rate
- We are so good at this that its hard to appreciate how difficult it is.
  - Its one of the main difficulties in making computers perceive.
  - We still don’t have generally accepted solutions.

The invariant feature approach

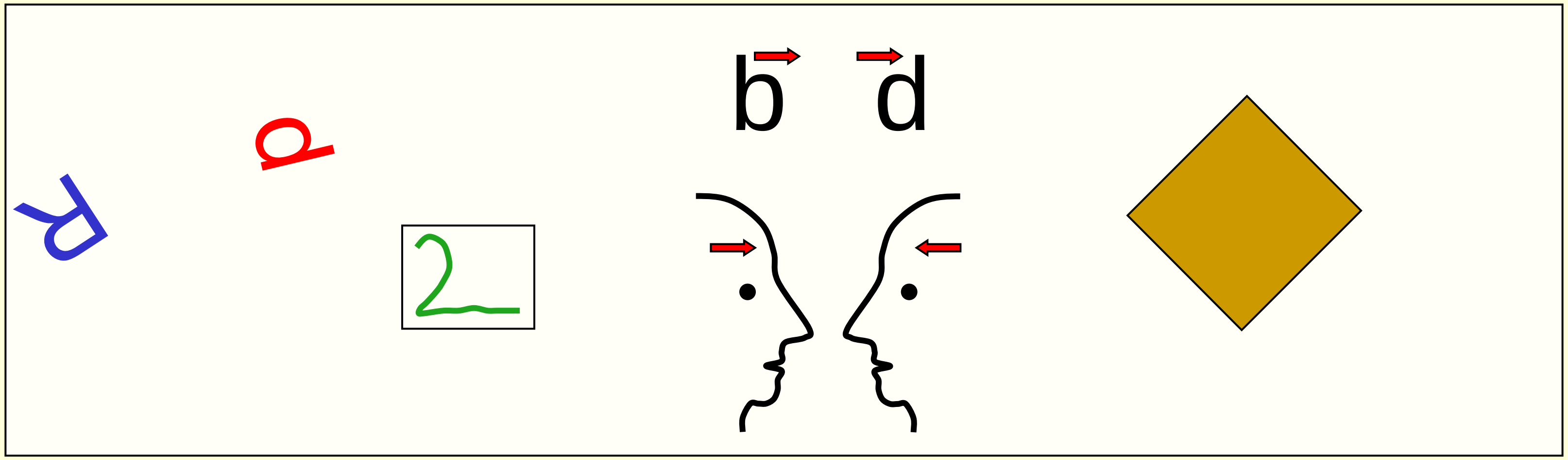
- Extract a large, redundant set of features that are invariant under transformations
  - e.g. “pair of parallel lines with a dot between them.



- With enough of these features, there is only one way to assemble them into an object.
  - we don’t need to represent the relationships between features directly because they are captured by other features.
- We must avoid forming features from parts of different objects!

The normalization approach

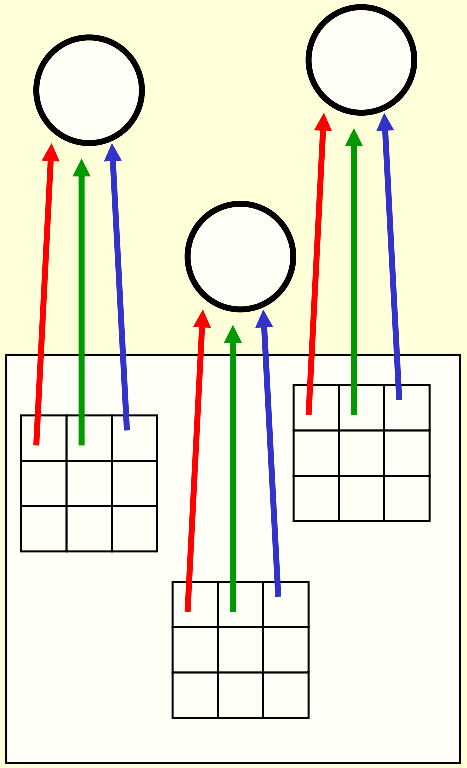
- Do preprocessing to normalize the data
  - e. g. put a box around an object and represent the locations of its pieces relative to this box.
  - Eliminates as many degrees of freedom as the box has.
    - translation, rotation, scale, shear, elongation
  - But its not always easy to choose the box



The replicated feature approach

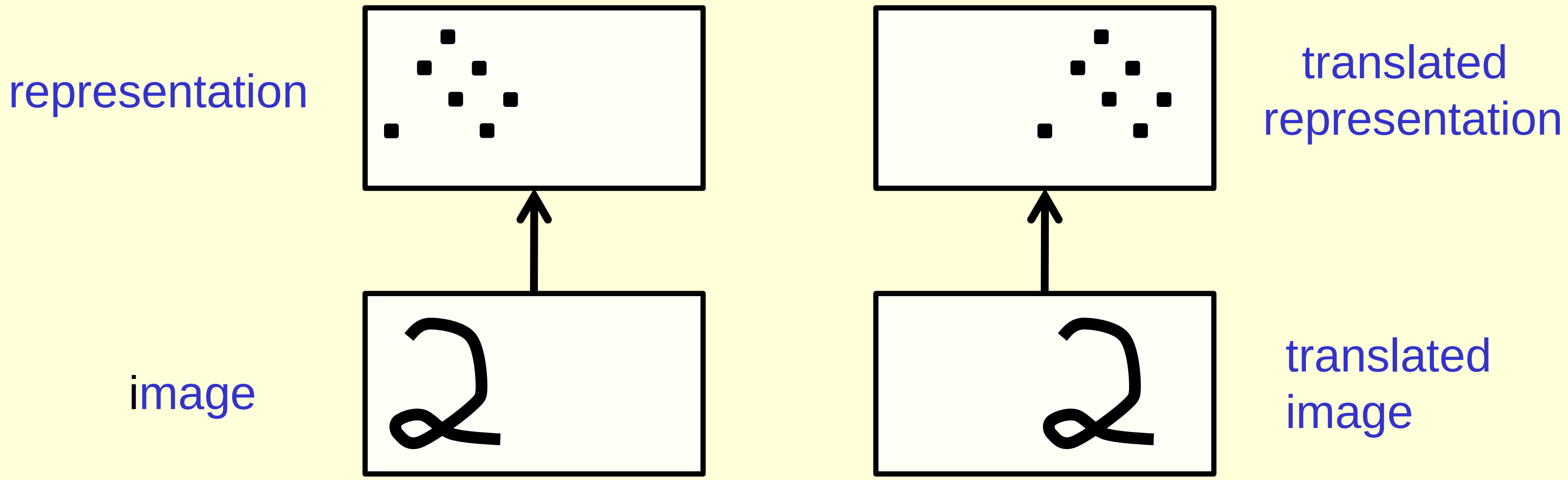
- Use many different copies of the same feature detector.
  - The copies all have slightly different positions.
  - Could also replicate across scale and orientation.
    - Tricky and expensive
  - Replication reduces the number of free parameters to be learned.
- Use several different feature types, each with its own replicated pool of detectors.
  - Allows each patch of image to be represented in several ways.

The red connections all have the same weight.



Equivariance

- Without the sub-sampling, convolutional neural nets give "equivariance" for discrete translations.



- A small amount of translational invariance can be achieved at each layer by using local averaging or maxing.
  - This is called "sub-sampling".

Backpropagation with weight constraints

- It is easy to modify the backpropagation algorithm to incorporate linear constraints between the weights.
- We compute the gradients as usual, and then modify the gradients so that they satisfy the constraints.
  - So if the weights started off satisfying the constraints, they will continue to satisfy them.

$$To\ \ constrain: \quad w_1 = w_2$$

$$we\ \ need: \quad \Delta w_1 = \Delta w_2$$

$$compute: \quad \frac{\partial E}{\partial w_1} \quad and \quad \frac{\partial E}{\partial w_2}$$

$$use \quad \frac{\partial E}{\partial w_1} + \frac{\partial E}{\partial w_2} \quad for\ w_1\ and\ w_2$$

Combining the outputs of replicated features

- Get a small amount of translational invariance at each level by averaging four neighboring replicated detectors to give a single output to the next level.
  - This reduces the number of inputs to the next layer of feature extraction, thus allowing us to have many more different feature pools.
  - Taking the maximum of the four works slightly better.

Combining the outputs of replicated features

- If you don’t understand anything about coarse coding, achieving invariance in multiple stages by alternating between a layer of feature extraction and a layer of sub-sampling seems to fit what we know about the monkey visual system.
  - The larger receptive fields of higher-level features suggest that they do not care about position.
  - But this naïve view cannot explain how we preserve the precise spatial relationship between fairly high-level features like a nose and a mouth.

Equivariance vs Invariance

- Sub-sampling tries to make the neural activities invariant for small changes in viewpoint.
  - This is probably the wrong goal.
  - It is motivated by the fact that the final label needs to be viewpoint-invariant.
- Its probably better to aim for equivariance:
  - Changes in viewpoint lead to corresponding changes in neural activities.

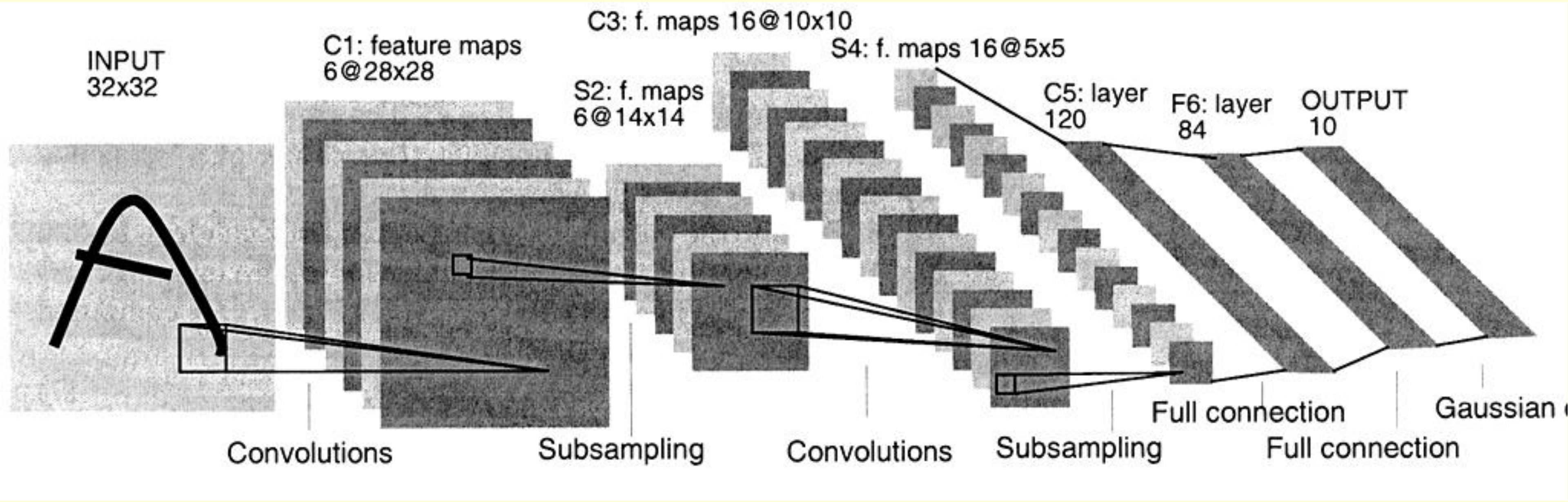
Equivariance vs Invariance

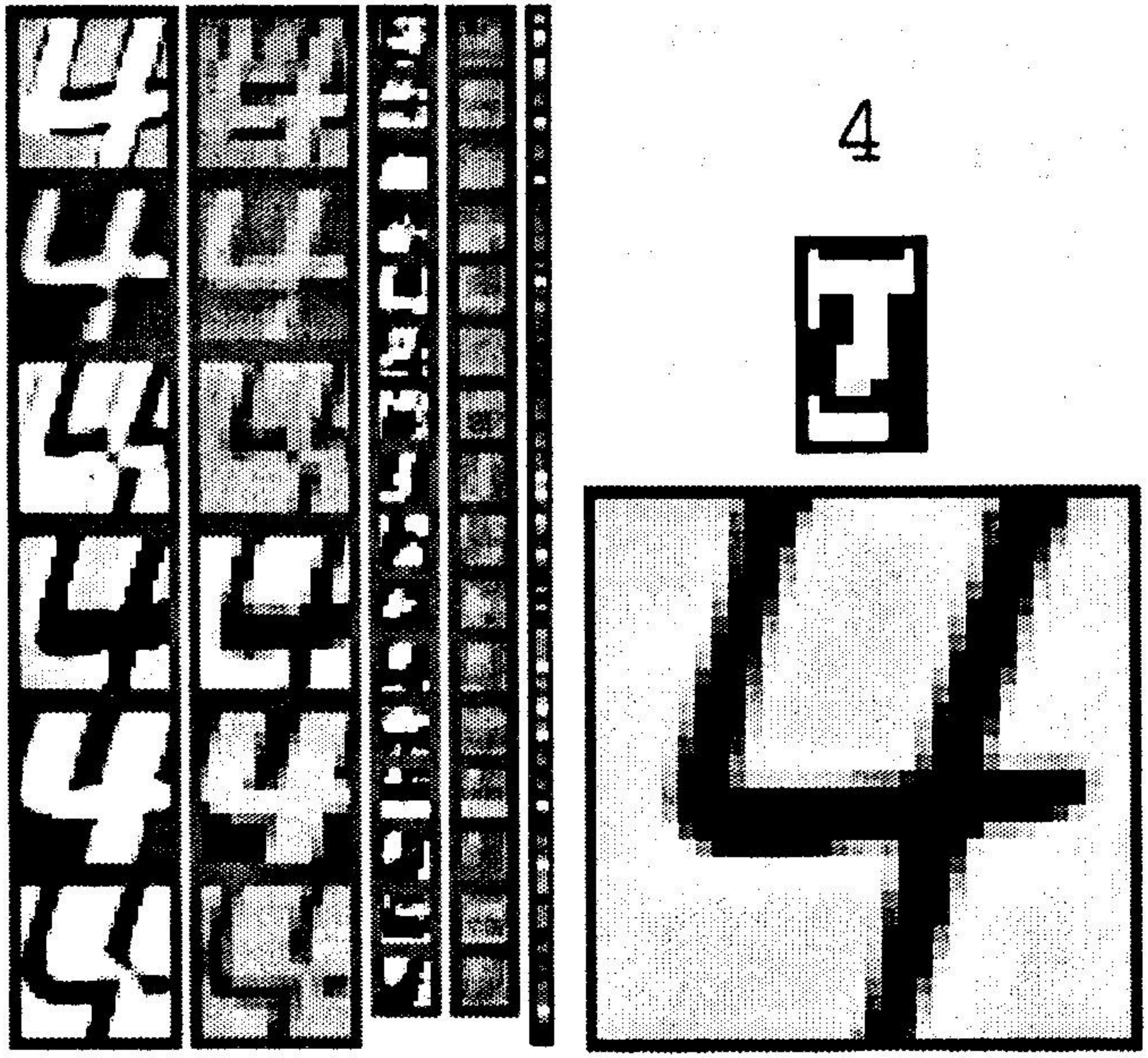
- In the perceptual system, its the weights that code viewpoint-invariant knowledge, not the neural activities.
  - Convolutional nets (without sub-sampling) achieve this by copying the weights to every location.
  - We need a better way of achieving the same thing that is more neurally plausible and works efficiently for more than just position invariance.
    - Dilation, rotation, elongation, shear, lighting etc.

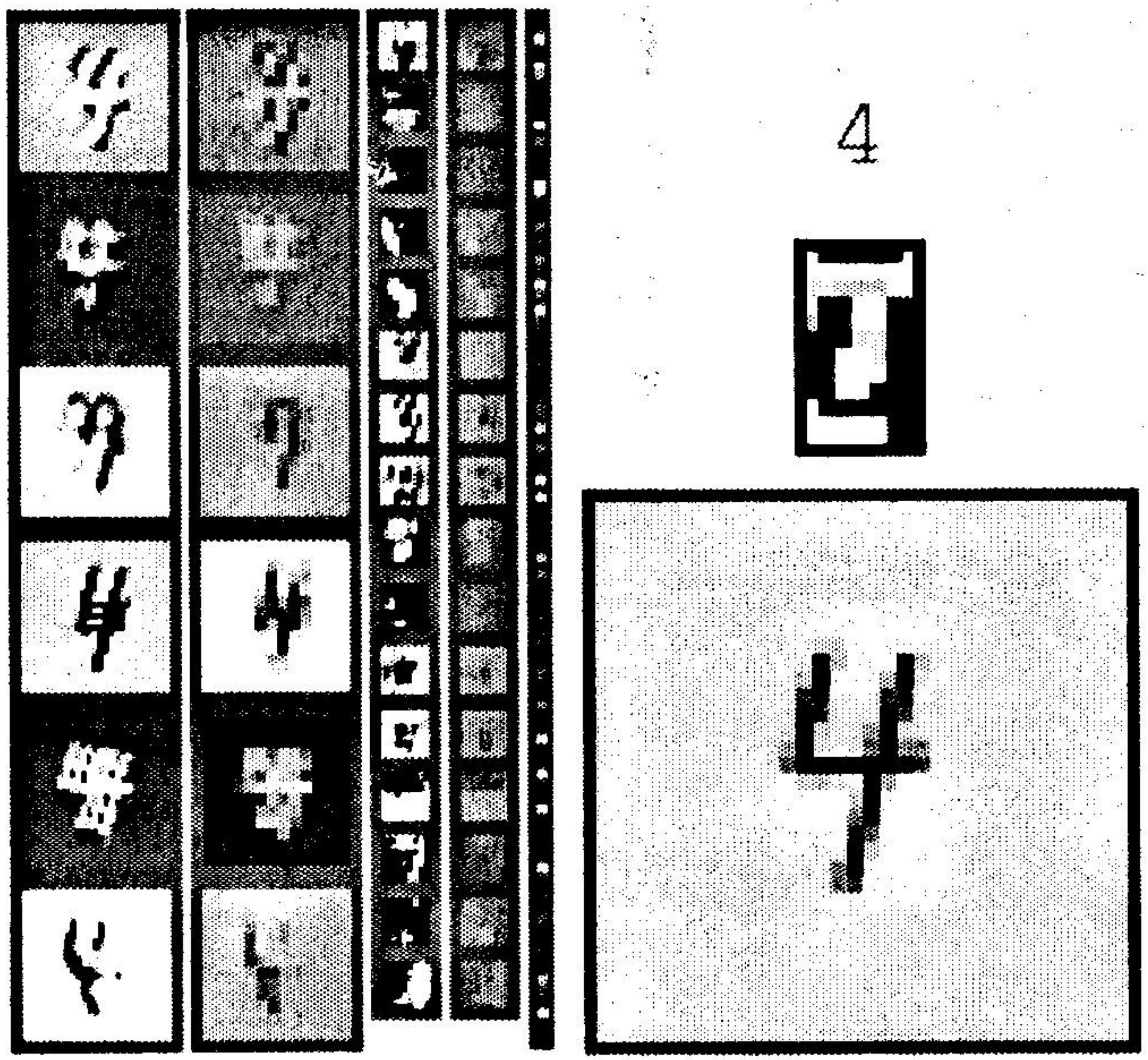
Le Net

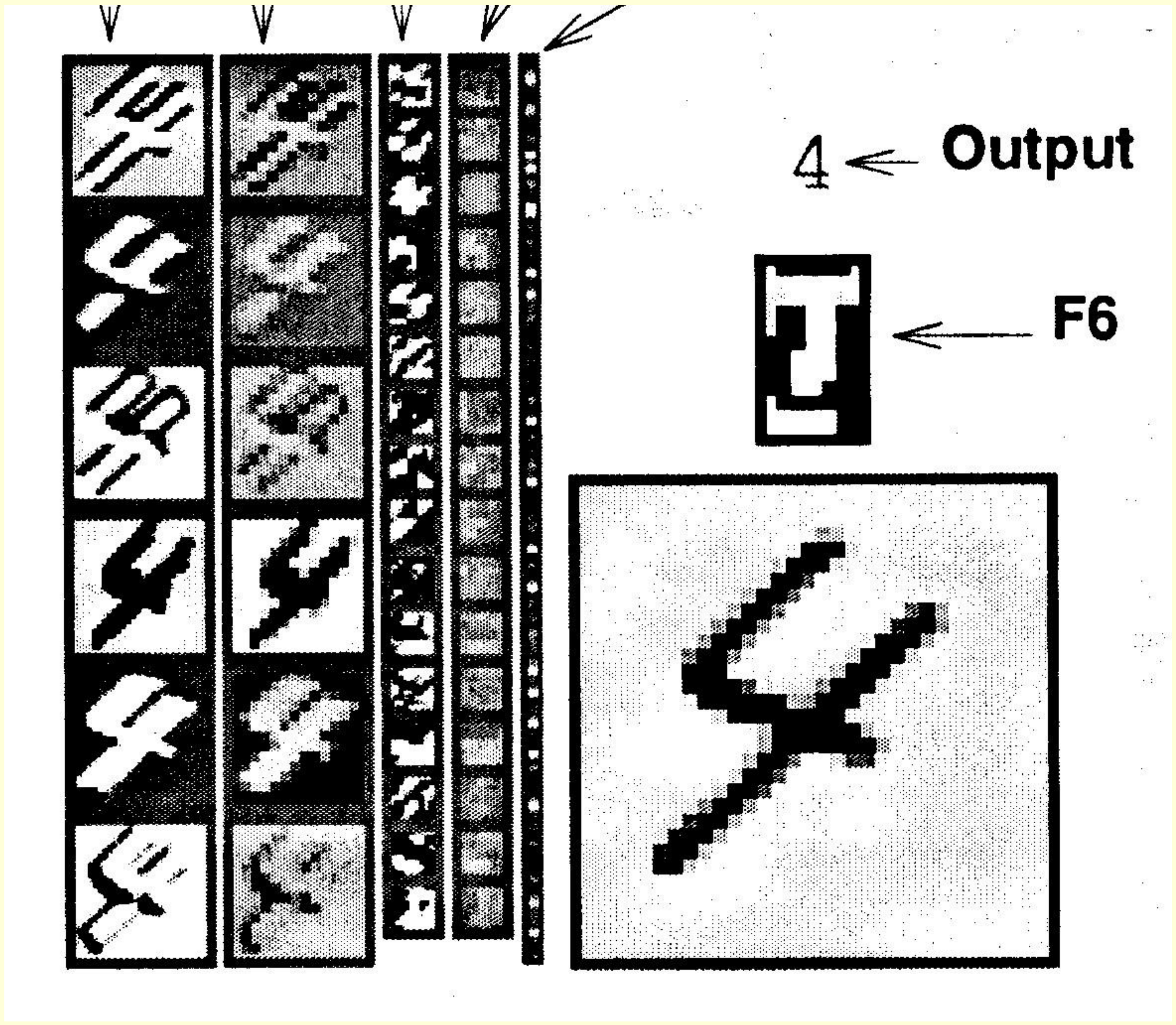
- Yann LeCun and others developed a really good recognizer for handwritten digits by using backpropagation in a feedforward net with:
  - Many hidden layers
  - Many pools of replicated units in each layer.
  - Averaging of the outputs of nearby replicated units.
  - A wide net that can cope with several characters at once even if they overlap.
- Look at all of the demos of LENET at http://yann.lecun.com
  - These demos are required "reading" for the tests.

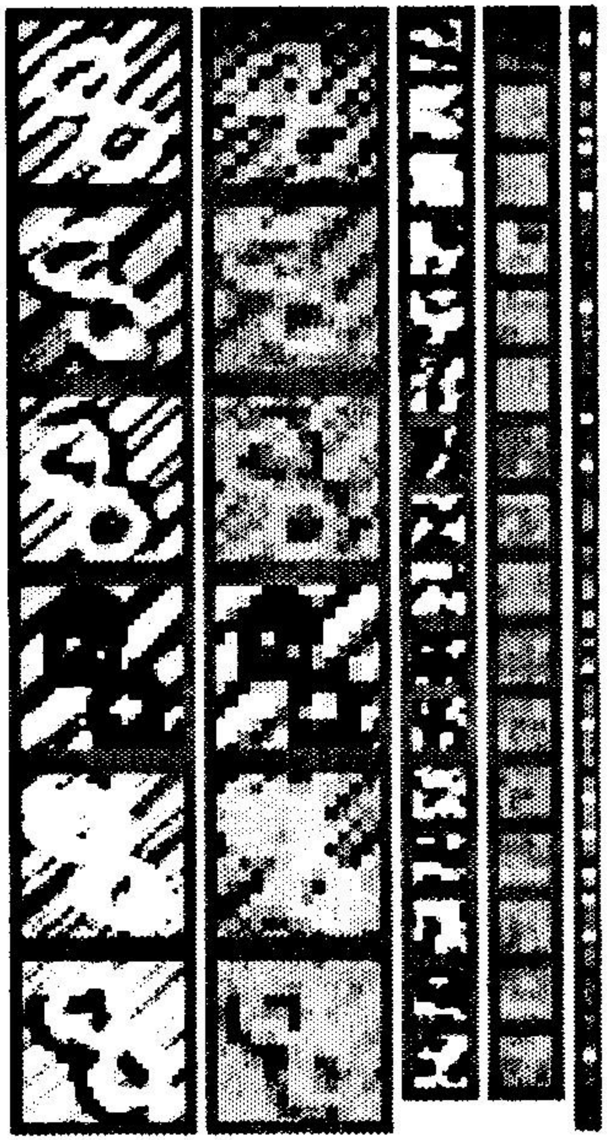
The architecture of LeNet5



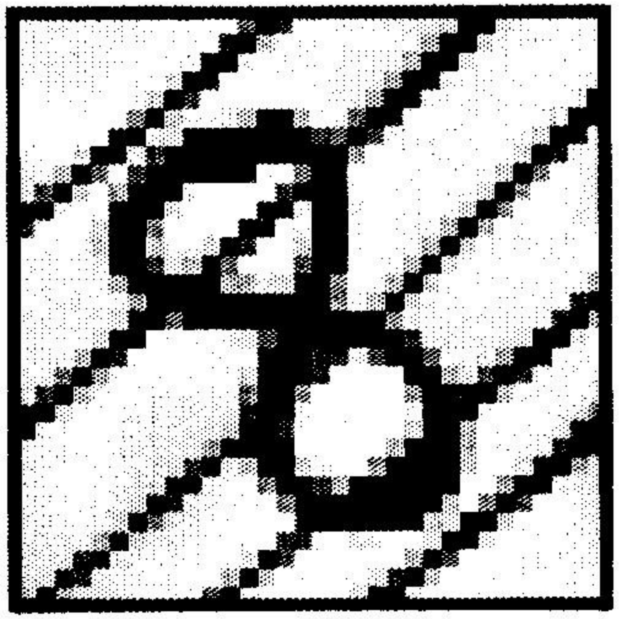

4

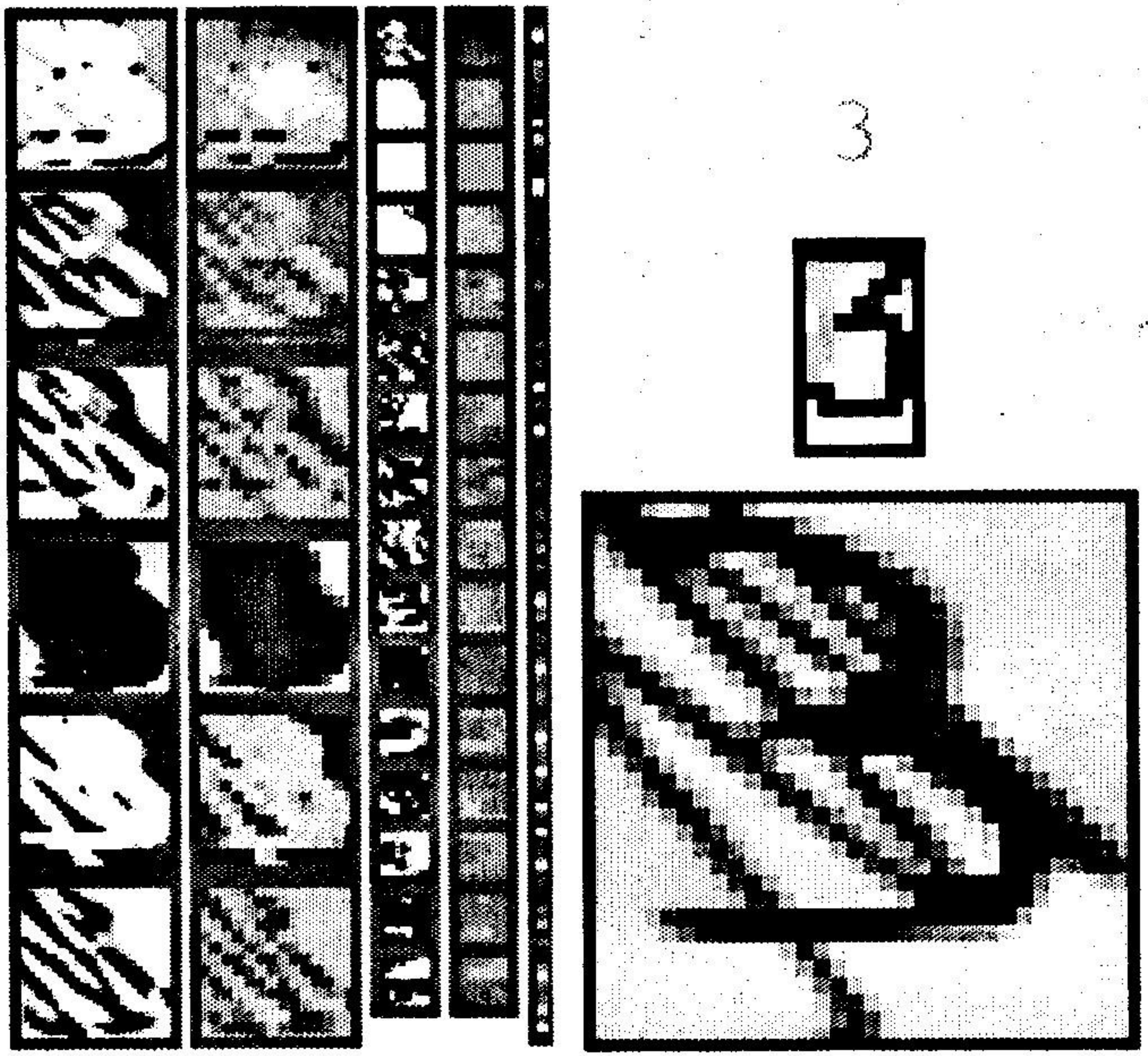

4

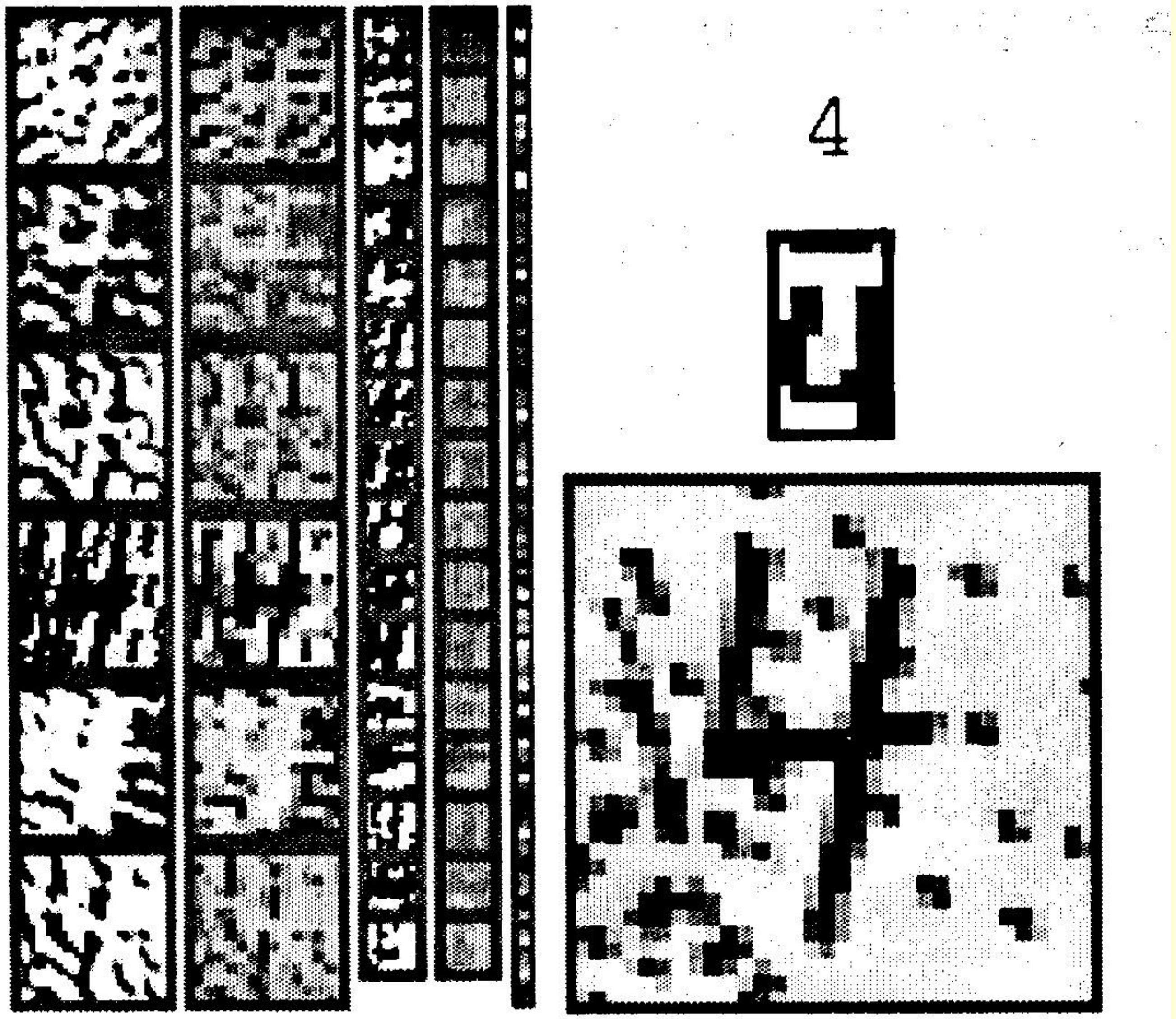

4
Output
F6

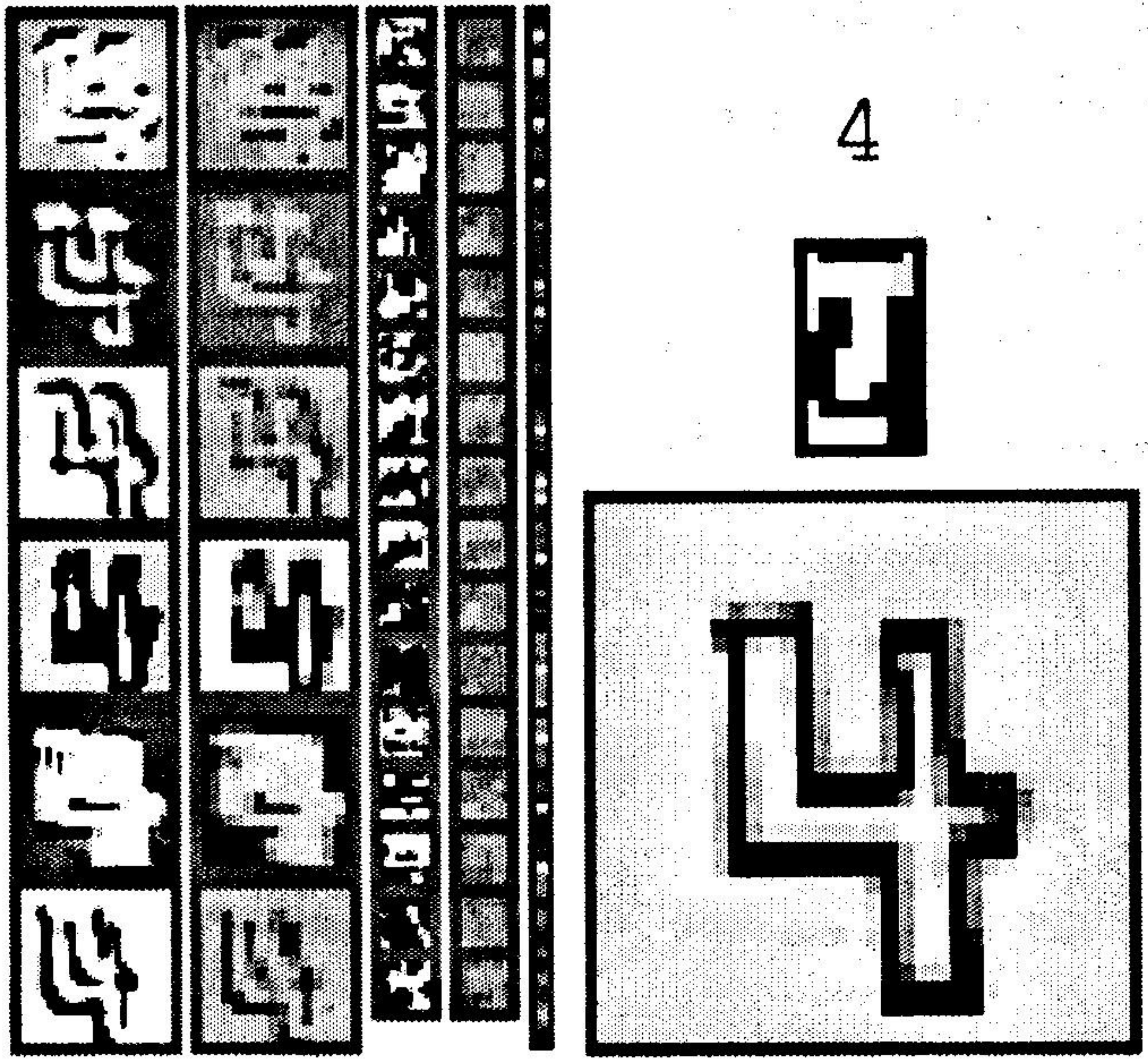


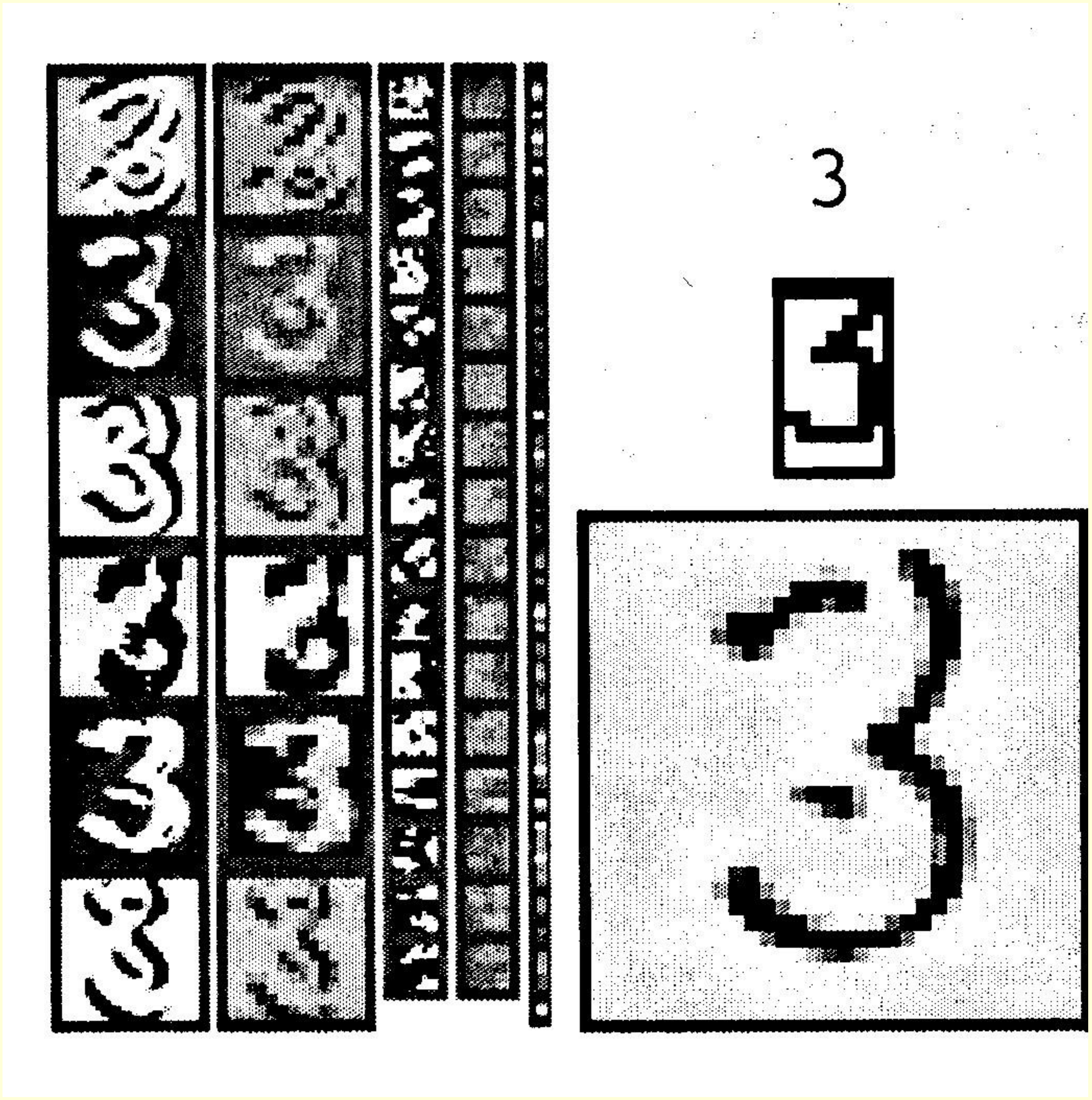
8

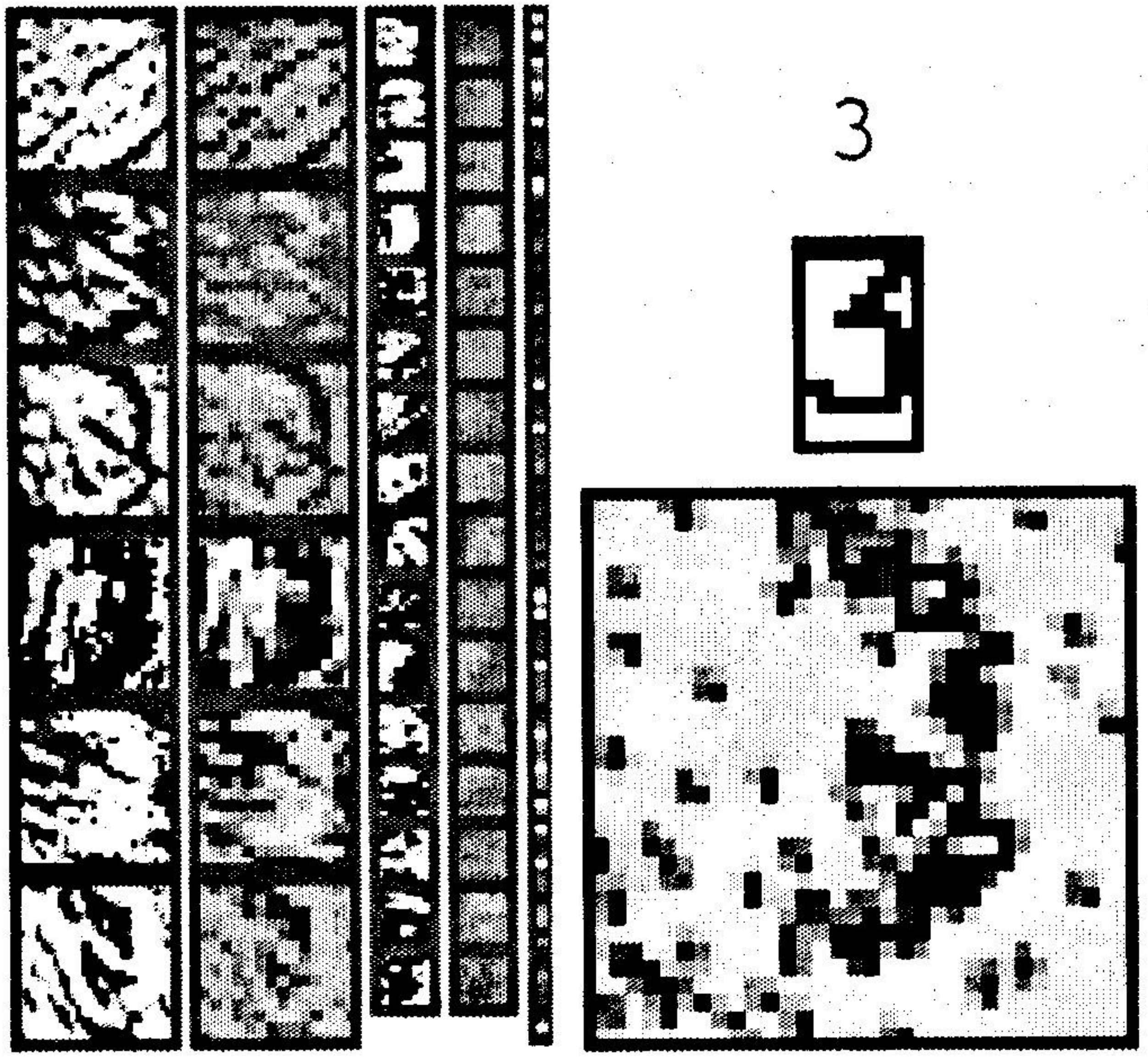



3


4


4


3


3

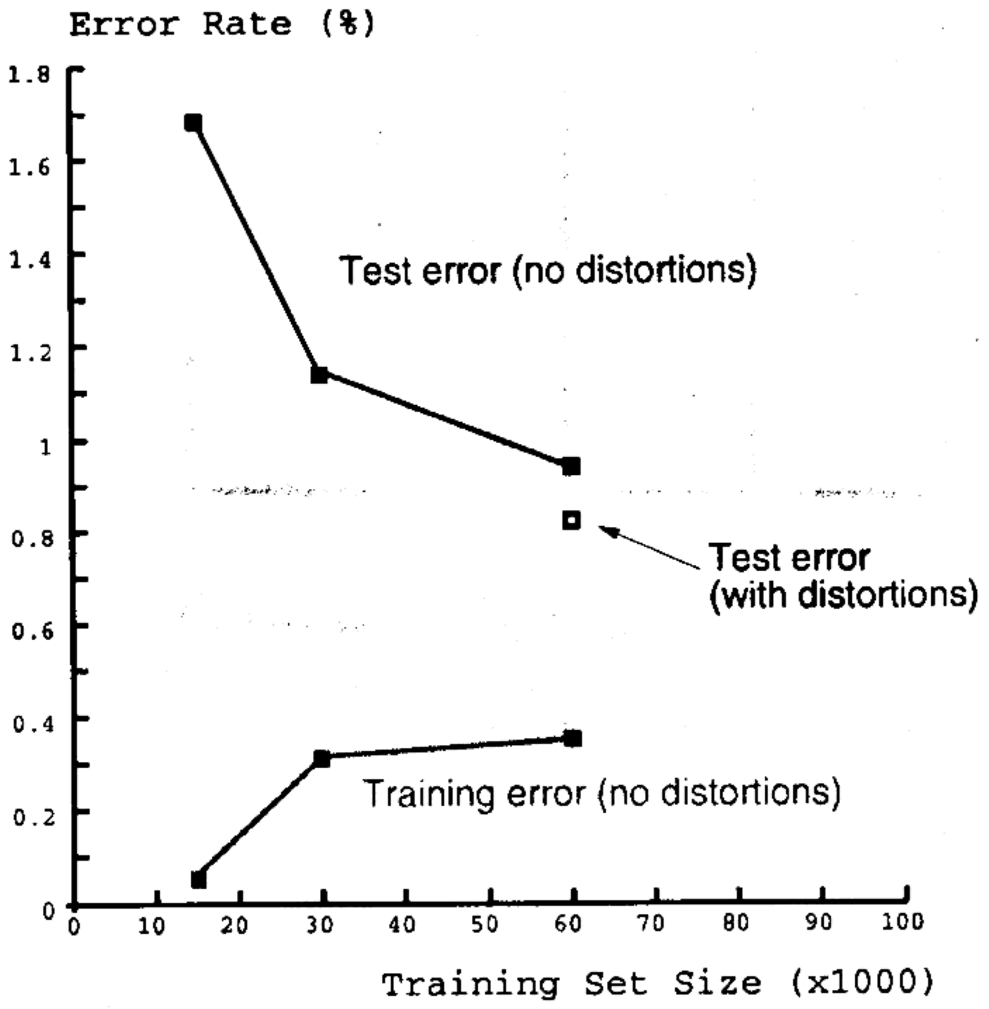
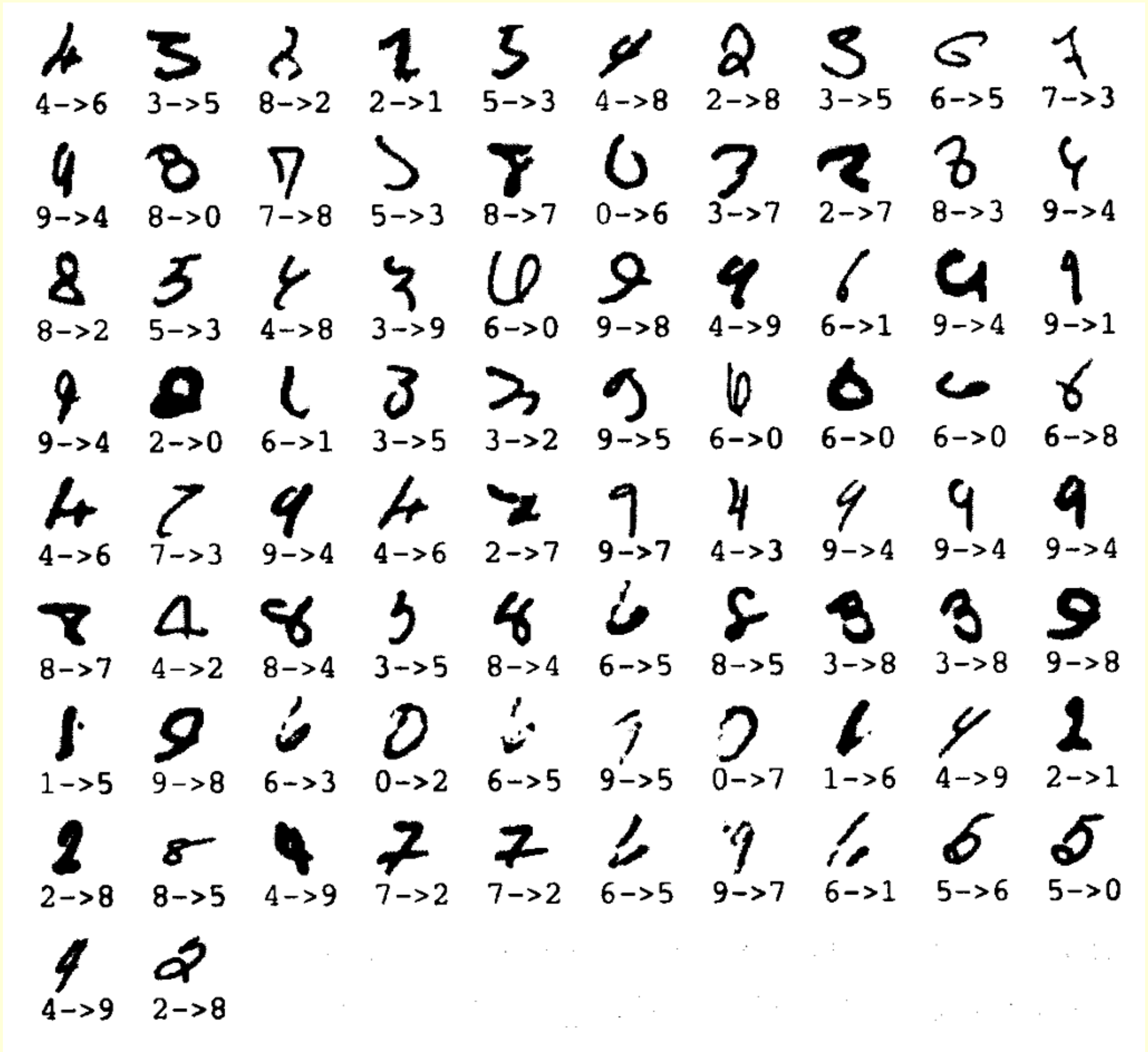


Fig. 6. Training and test errors of LeNet-5 achieved using training sets of various sizes. This graph suggests that a larger training set could improve the performance of LeNet-5. The hollow square show the test error when more training patterns are artificially generated using random distortions. The test patterns are not distorted.

The 82 errors made by LeNet5



Notice that most of the errors are cases that people find quite easy.

The human error rate is probably 20 to 30 errors

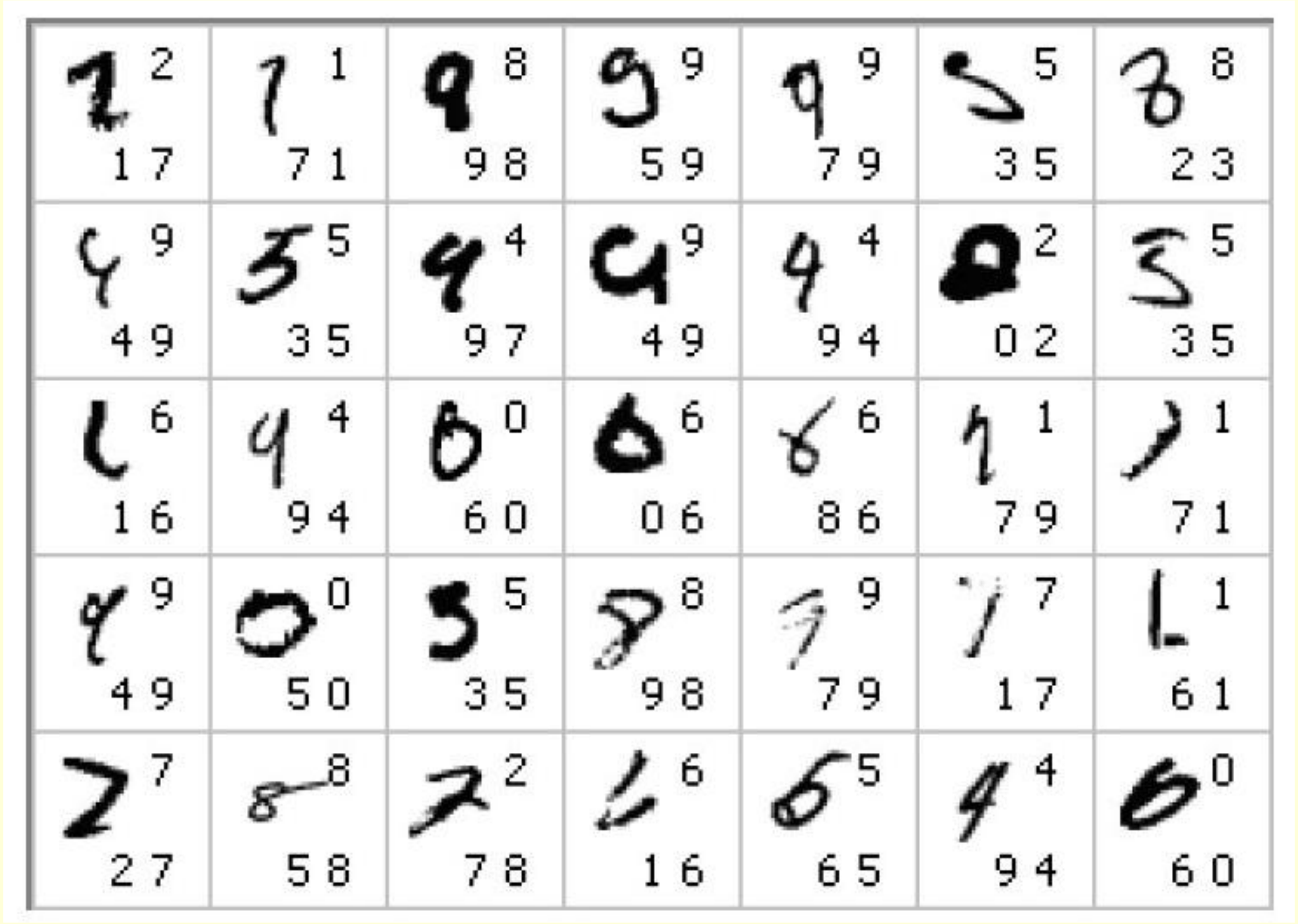
A brute force approach

- LeNet uses knowledge about the invariances to design:
  - the network architecture
  - or the weight constraints
  - or the types of feature
- But its much simpler to incorporate knowledge of invariances by just creating extra training data:
  - for each training image, produce new training data by applying all of the transformations we want to be insensitive to (Le Net can benefit from this too)
  - Then train a large, dumb net on a fast computer.
  - This works surprisingly well if the transformations are not too big (so do approximate normalization first).

Making dumb backpropagation work really well for recognizing digits (Ciresan et. al. 2010)

- Using the standard viewing transformations plus local deformation fields to get LOTS of data.
- Use a large number of hidden layers with a large number of units per layer and no weight constraints.
- Use the appropriate error measure for multi-class categorization:
  - Softmax outputs with cross-entropy error.
- Train the hell out of it by using a big GPU board for a long time (at 5x10^9 weight updates per second)
  - This gets down to only 35 errors which is the record and is close to human performance.

The errors made by the big dumb net trained with lots of fancy transformations of the data



The top printed digit is the right answer. The bottom two printed digits are the network's best two guesses.

Priors and Prejudice

- We can put our prior knowledge about the task into the network by using weight-sharing, or carefully designing the connectivity, or carefully choosing the right types of unit.
  - But this prejudices the network in favor of a particular way of solving the problem.
- Alternatively, we can use our prior knowledge to create a whole lot more training data.
  - This may require a lot of work and it is much less efficient in terms of the time required for learning.
  - But it does not prejudice the network in favor of a particular way of getting the right answer.