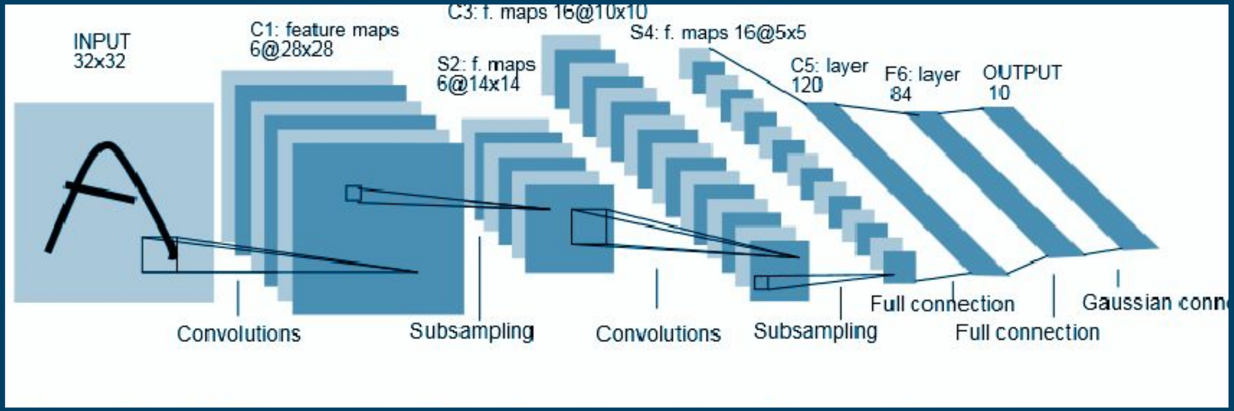
# Gradient-Based Learning Applied To Document Recognition



**Group 07**

E/15/076-Sandushi

E/15/077-Kshithija

E/15/211-Ishani

E/15/279-Wathsari

# BACKGROUND

## The Authors

- Yann LeCun
- Leon Bottou
- Yoshua Bengio
- Patrick Haffner

## Presented in

- Proceedings Of The IEEE journal

## in

- 1998

# INTRODUCTION

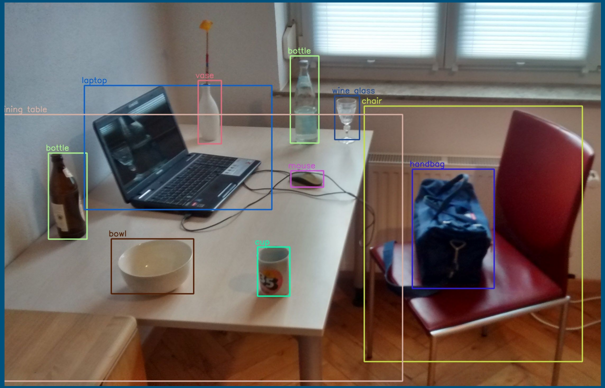
- Why Gradient based learning?
  - easier to minimize a reasonably smooth,continuous function than  a discrete function!
- backpropagation
- Learning In Real Handwriting Recognition Systems
  - Heuristic Over Segmentation
- Globally Trainable Systems
  - Most practical pattern recognition systems are composed of multiple modules.

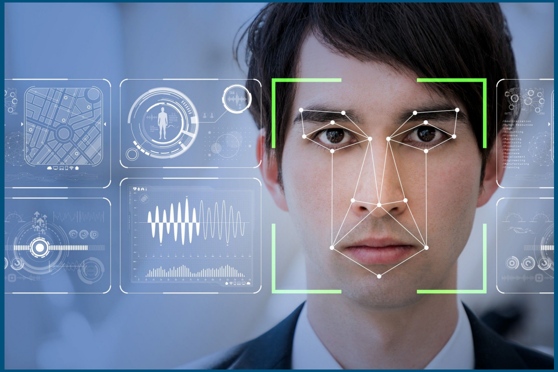
# CONVOLUTIONAL NEURAL NETWORKS FOR ISOLATED CHARACTER RECOGNITION

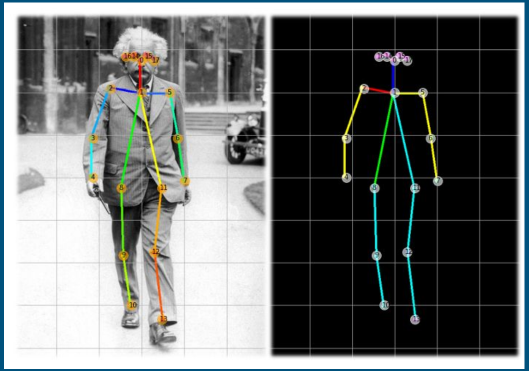
# Convolutional Networks

- Convolutional Neural Networks is the standard form of neural network architecture for solving tasks associated with images.

- Solutions for tasks such as object detection, face detection, pose estimation and more all have CNN architecture variants.







- A few characteristics of the CNN architecture makes them more favourable in several computer vision tasks.
    - Local Receptive Fields
    - Sub-Sampling
    - Weight Sharing

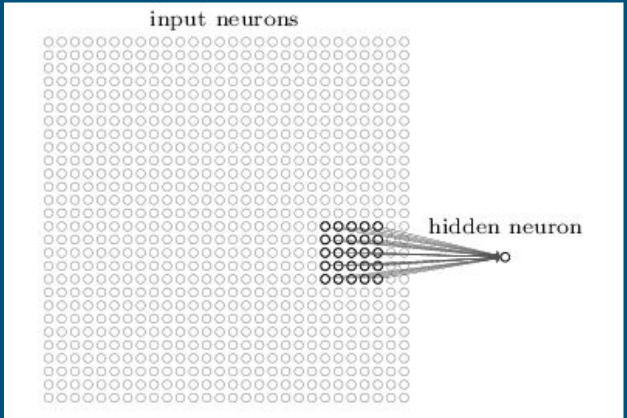


Fig01

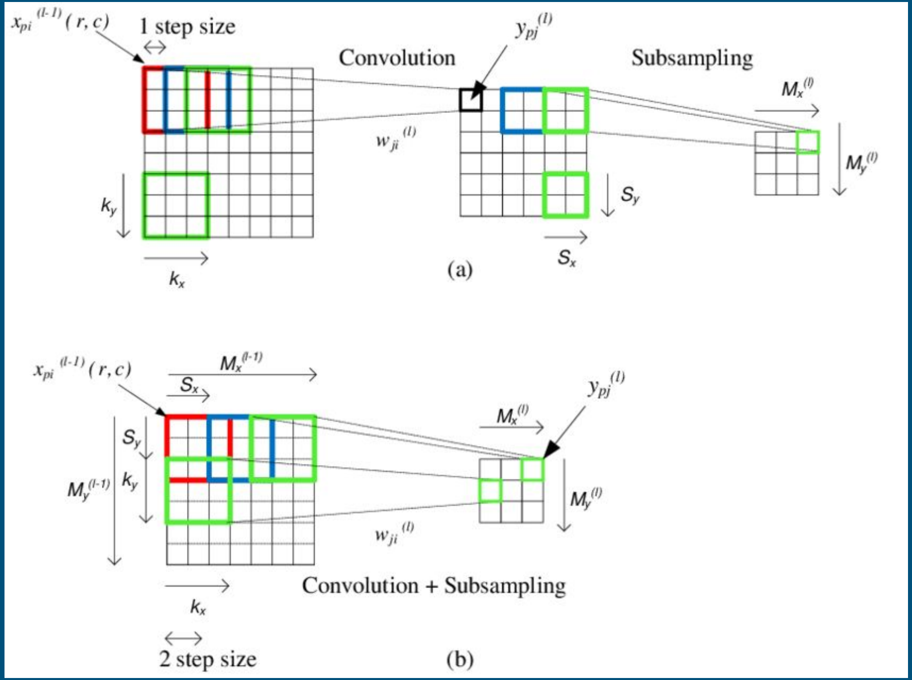


Fig02

# LeNet-5

- LeNet-5 CNN architecture is made up of 7 layers.
  - -3 convolutional layers
  - -2 subsampling layers
  - -2 fully connected layers

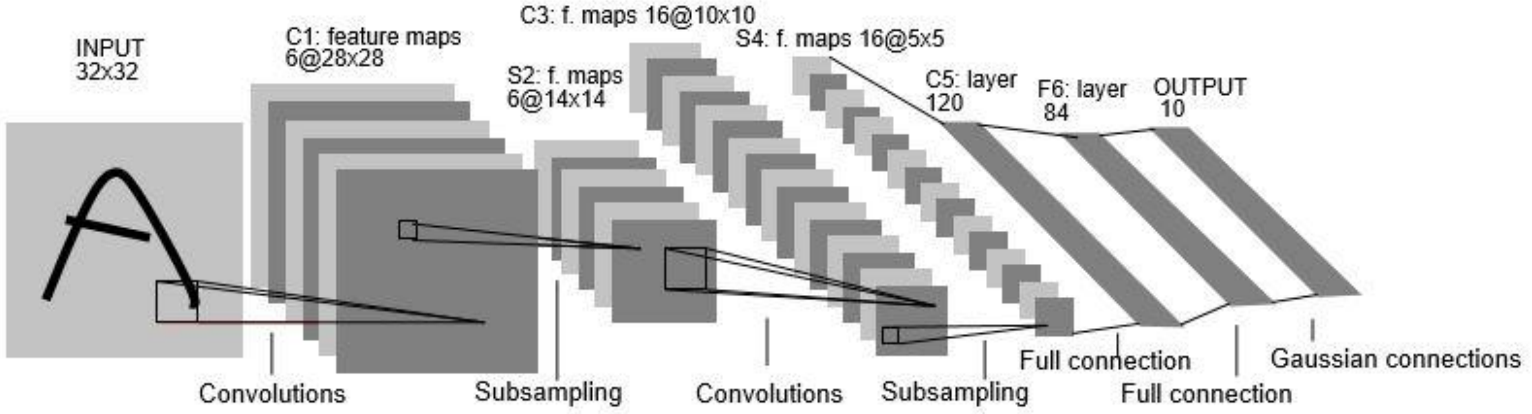


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- The LeNet5 construct two significant types of layer construct
  -Convolutional Layers
  -Sub-Sampling Layers

## Layer C1

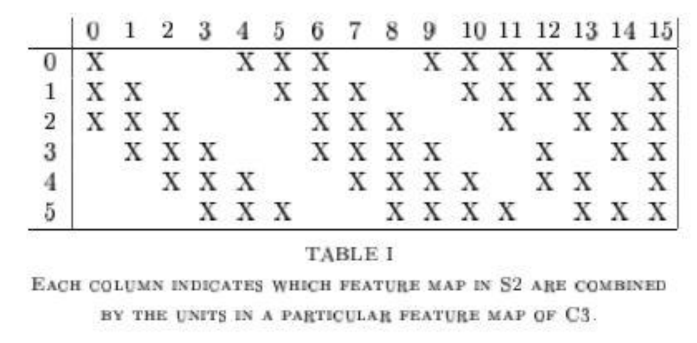
-A convolutional layer with 6 feature maps of size 28x28
-The size of the feature maps prevents connections from the input from falling off the boundary
-C1 contains 156 trainable parameters and 122,304 connections.

## Layer S2

-A sub-sampling  layer with 6 feature maps of size 14x14
-Each unit in each feature map is connected to a 2x2 neighbourhood in the corresponding feature map in C1
-do downsampling

## Layer C3

-A convolutional layer with 16 feature maps of size 28x28
-Each unit is connected to several 5x5 neighbourhoods st identical locations



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | X | | | | X | X | X | | | X | X | X | X | | X | X |
| 1 | X | X | | | | X | X | X | | | X | X | X | X | | X |
| 2 | X | X | X | | | | X | X | X | | | X | | X | X | X |
| 3 | | X | X | X | | | X | X | X | X | | | X | | X | X |
| 4 | | | X | X | X | | | X | X | X | X | | X | X | | X |
| 5 | | | | X | X | X | | | X | X | X | X | | X | X | X |

TABLE I

Each column indicates which feature map in S2 are combined by the units in a particular feature map of C3.

## Layer S4

-A sub-sampling layer with 16 feature maps of size 5x5
-Each unit in each feature map is connected in a similar way as C1 and S2
-S4 contains 32 trainable parameters and 2000 connections.

## Layer C5

-A convolutional layer with 120 feature maps of size 1x1
-S4 contains 48,120 trainable connections.

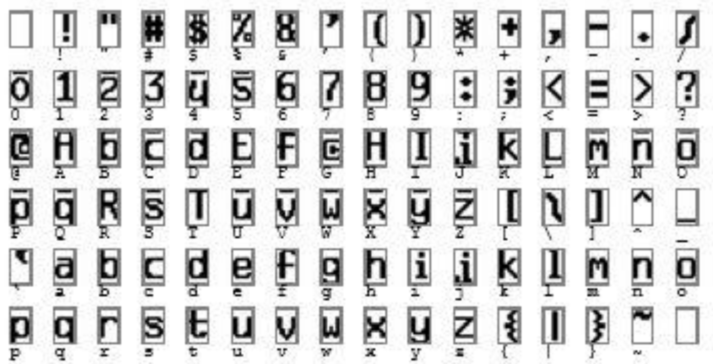


Fig. 3. Initial parameters of the output RBFs for recognizing the full ASCII set.

# Loss Function

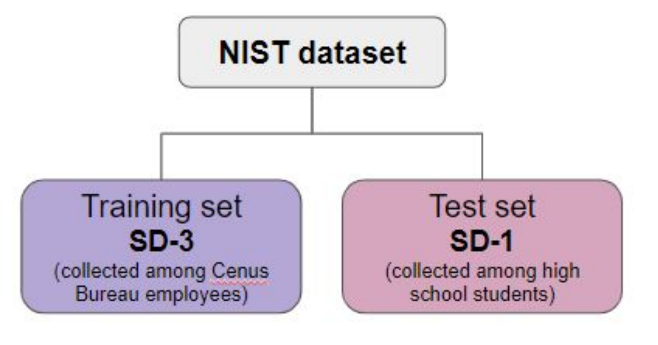
- The Loss is used to calculate the gradients. And gradients are used to update the weights of the Neural Net. This is how a Neural Net is trained.

$$E(W) = \frac{1}{P} \sum_{P=1}^{P} y D^{p}(Z^{p}, W)$$

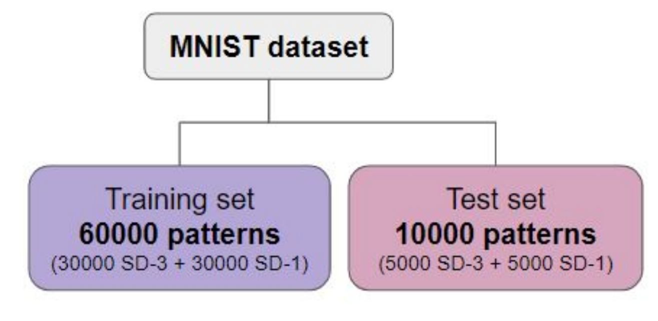
# RESULTS AND COMPARISON WITH OTHER METHODS

# A. Database : The Modified NIST set

- NIST - National Institute of Standards and Technology database
- MNIST database
  - of handwritten digits
  - Subset of NIST dataset
- Original images
  - Black & white
  - 20 x 20 pixel box



NIST dataset
Training set
SD-3
(collected among Cenus Bureau employees)
Test set
SD-1
(collected among high school students)



# A. Database cont.

- Three versions of database
  - First version - regular database
    - Centered in a 28 x 28 image
  - Second version - deslanting database
    - Deslanted and cropped down to 20 x 20 pixel
  - Third version
    - Reduced to 16 x 16 pixels

# B. Results

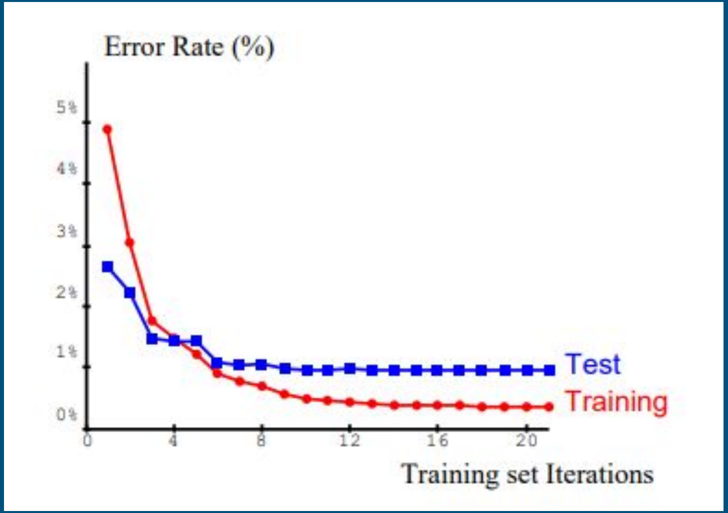
- LeNet-5
  - 20 iterations
  - Diagonal hessian approximation was re-evaluated on 500 samples
  - $\mu$ is a hand-picked constant ($\mu$ = 0.02 )
  - $h_{kk}$ is is an estimate of the second derivative of the loss function with respect to $\omega_k$
  - $\eta$ - global learning rate

$$\epsilon_k = \frac{\eta}{\mu + h_{kk}}$$

| Iteration | $\eta$ |
|---|---|
| 1-2 | 0.0005 |
| 3-5 | 0.0002 |
| 6-8 | 0.0001 |
| 9-12 | 0.00005 |
| 13=20 | 0.00001 |

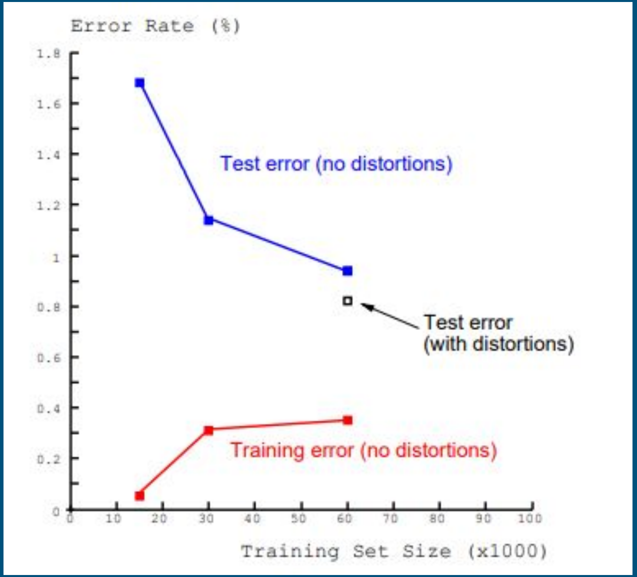
# Results cont.

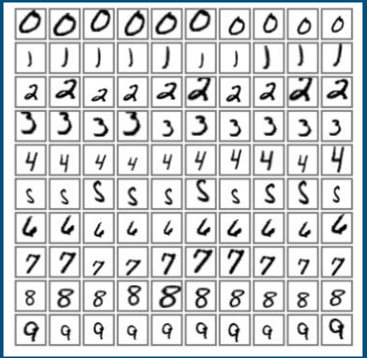
- Common phenomenon
  - When overtraining occurs, the training error keeps decreasing over time but the test error goes through a minimum and starts increasing after a certain number of iterations
- **It was not observed in this case**
  - Because learning rate was kept relatively large.
- The test error rate through the training set at 0.95%. (10 iterations)
- The error rate on the training set 0.35% (19 iterations)



# Results cont.

- Influence of training set
  - artificially generated more training examples
    - 60 000 original patterns
    - 540 000 instances of distorted patterns
      - horizontal and vertical translations
      - Scaling
      - Squeezing
  - Test error rate dropped to 0.8% (from 0.95%)



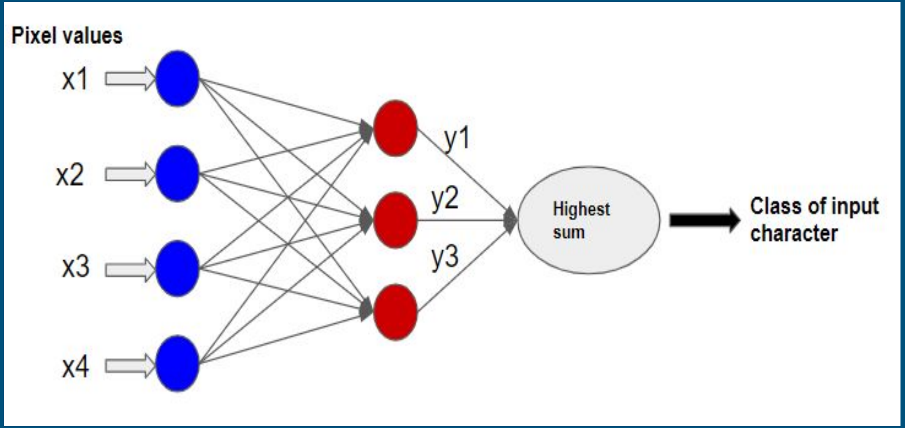


# C. Comparison with Other Classifiers

## C.1.Linear Classifier and Pairwise Linear Classifier

- Simplest classifier
- Each input pixel value contributes to a weighted sum for each output unit. The output unit with the highest sum (including the contribution of a bias constant) indicates the class of the input character.
- simple improvement of the basic linear classifier
  - train each unit of a single layer network to separate each class from each other class

| Database | Error rate | Free parameters |
|---|---|---|
| Regular data | 12% | 7850 |
| Deslanted images | 8.4% | 4010 |



# Comparison with Other Classifiers cont.

## C.2. Baseline Nearest Neighbour Classifier

- A K-NN classifier with a Euclidean distance measure between input images
  - No training time
  - No thought on the part of the designer
  - But, memory requirement and recognition time are large
  - 60000 , 20 x 20 pixel images
- K = 3

| Database | Error rate |
|---|---|
| Regular data | 5% |
| Deslanted images | 2.4% |

# Comparison with Other Classifiers cont.

## C.3.Principle Component Analysis(PCA) and Polynomial Classifier

- To compute the principal components
  - First, compute the mean of each input component
  - Second, subtract from the training vectors
  - Next, compute covariance matrix of the resulting vectors
  - Then, diagonalize using singular value decomposition
- Second degree polynomial classifier
  - Input - 40-dimensional feature vector
  - A linear classifier with 821 inputs

| Database | Error rate |
|---|---|
| Regular data | 3.3% |

# Comparison with Other Classifiers cont.

## C.4.Radial Basis Function Network

- An RBF network
  - First layer - 1000 Gaussian RBF units with 28 28 inputs
  - Second layer - simple 1000 inputs/ten outputs linear classifier
  - RBF units were divided into ten groups of 100
    - using the adaptive K-means algorithm
  - Second-layer weights
    - compute using a regularized pseudo inverse method

| Database | Error rate |
|---|---|
| Regular data | 3.6% |

# Comparison with Other Classifiers cont.

## C.5.One-Hidden Layer Fully Connected MultiLayer Neural Network

- A fully connected multilayer NN
  - Two layers of weights
  - Trained with the version of back-propagation

| Database | Error rate | No.of hidden units |
|---|---|---|
| Regular data | 4.7% | 300 |
| | 4.5% | 1000 |
| Deslanted images | 1.6% | 300 |

# Comparison with Other Classifiers cont.

## C.6.Two-Hidden Layer Fully Connected MultiLayer Neural Network

- A two-hidden-layer multilayer NN
  - A much better result than the one-hidden-layer network

| Network | Error rate |
|---|---|
| 28x28- 300-100-10 network | 3.05% |
| 28x28-1000-150-10 network | 2.95% |
| 28x28- 300-100-10 network | 2.50% |
| 28x28-1000-150-10 network | 2.45% |

# Comparison with Other Classifiers cont.

## C.7.A Small Convolutional Network:LeNet-1

- LeNet-1
  - For comparison purposes
  - Images - 16x16 pixels & centered in the 28 28 input layer
  - Develop- own version of the USPS (U.S. Postal Service zip codes) database
  - Test error - 1.7%

# Comparison with Other Classifiers cont.

## C.8.LeNet-4

- LeNet-4
    - For  large size of the training set
    - Test error - 1.1%
    - Replace the last layer -  a Euclidean nearest-neighbor classifier
    - Improve rejection performance

# Comparison with Other Classifiers cont.

## C.9.Boosted LeNet-4

- Three LeNet-4
  - First - train usually
  - Second - train on patterns that are filtered by the first net
  - Third - train on new patterns
- Test error rate - 0.7%

# Comparison with Other Classifiers cont.

## C.10.Tangent Distance Classifier(TDC)

- Tangent distance classifier - a nearest-neighbor method
- Test error rate - 1.1%
  - Using 16x16 pixel images

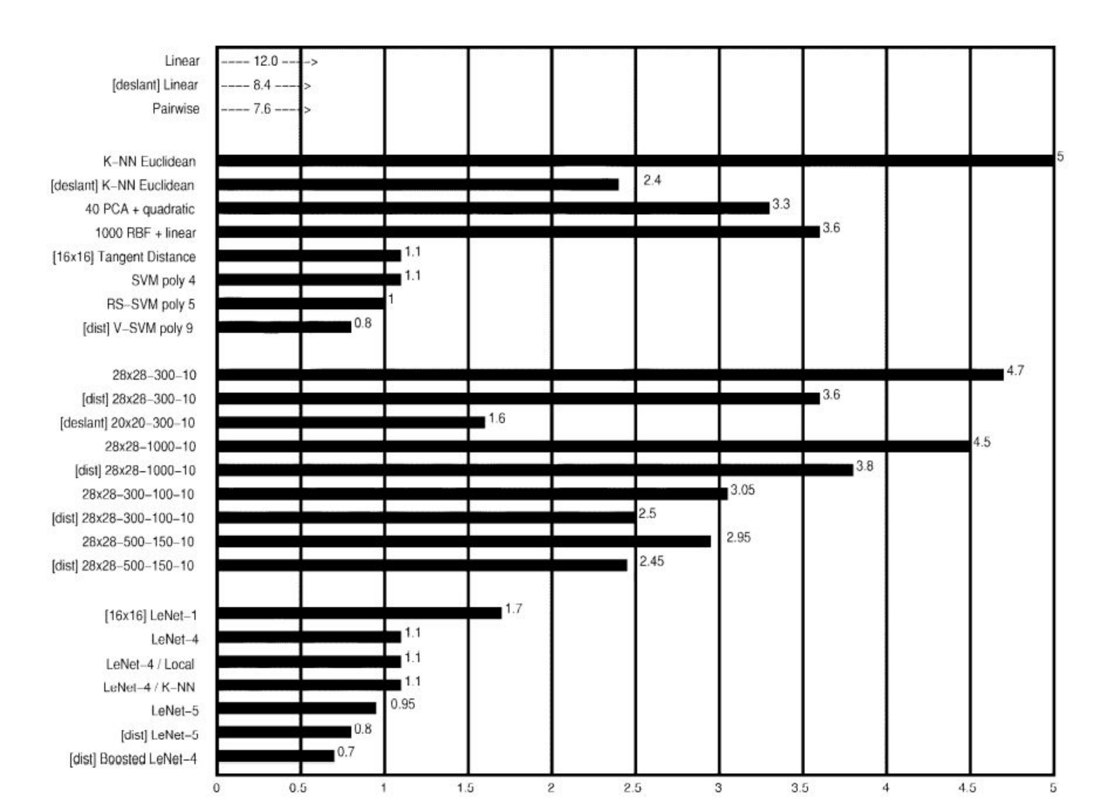
# Comparison with Other Classifiers cont.

## C.11.Support Vector Machine(SVM)

| Method | Error rate |
|---|---|
| Regular SVM | 1.4% |
| Scholkopf’s V-SVM | 1.0% |
| modified V-SVM | 0.8% |
| Burges’s RS-SVM technique | 1.1% |

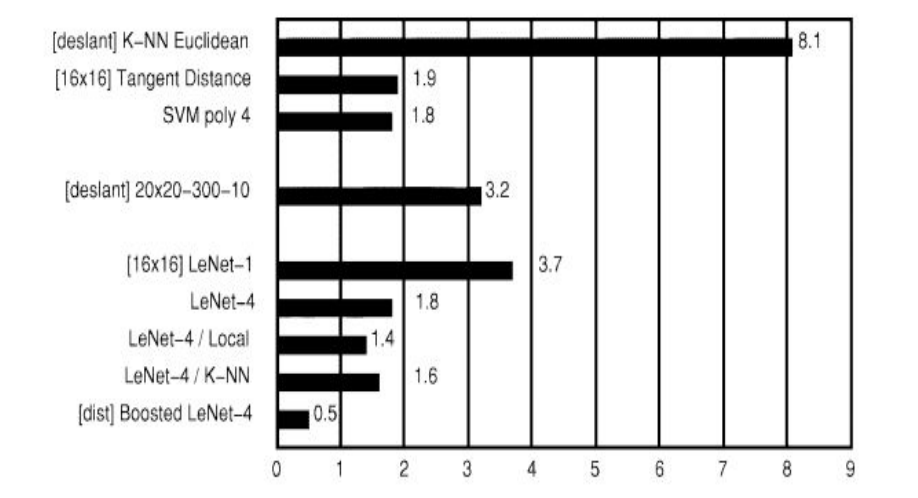
# Discussion

- Raw error rate of the Classifiers on the 10 000 example test set.
- Boosted LeNet-4 performed best (0.7%)
- Closely followed by LeNet-5 at 0.8%.



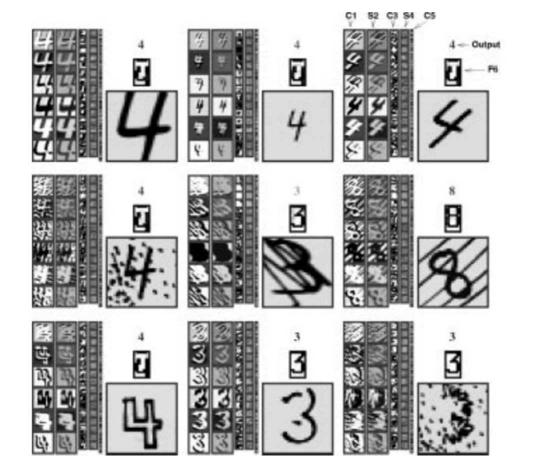
# Discussion cont.

- Rejection performance
  - Again, Boosted LeNet-4 has the best performance.
  - The enhanced versions of LeNet4 did better than the original LeNet-4
- Memory requirements
  - Most methods - 1 byte , nearest neighbor methods- 4 bits per pixel
- **Boosting gives a substantial improvement in accuracy, SVM has excellent accuracy.**



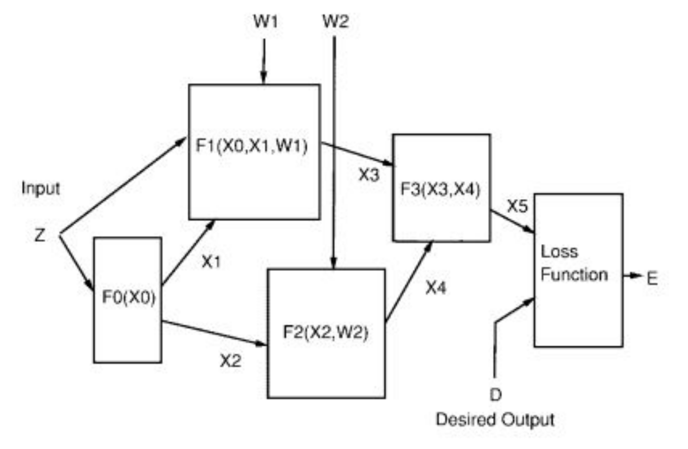
# Invariance and Noise Resistance

- Important is not obvious
- Upper and lower profiles of entire fields are detected and used to normalize the image to a fixed height.
- MNIST training set with salt and pepper noise
- Each pixel - randomly inverted with probability 0.1 $^2$.



# MULTI-MODULE SYSTEMS AND GRAPH TRANSFORMER NETWORKS

- Train systems composed of multiple heterogeneous modules?
  - large and complex trainable systems need to be built out of simple, specialized modules
    - LeNet-5
    - system for recognizing words



# A. An Object-Oriented Approach

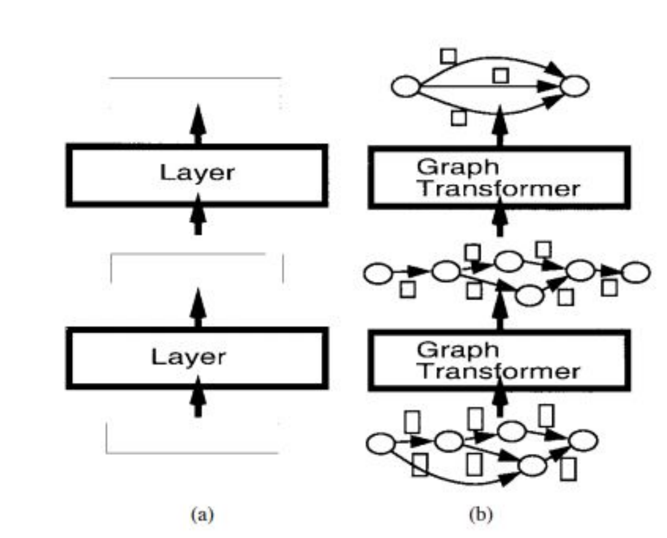
- Multi-module systems
    - Each module is an instance of a class
    - Module classes have a fprop method
    - bprop method for computing derivatives
    - Duality property between fprop & bprop
- SN3.1
    - Software environment used to obtain the results
    - Based on a home-grown object-oriented dialect

# B. Special Modules

- Multiplexer module
  - Two (or more) regular inputs,
  - One switching input
  - One output
- Min module
  - Two inputs
  - One output

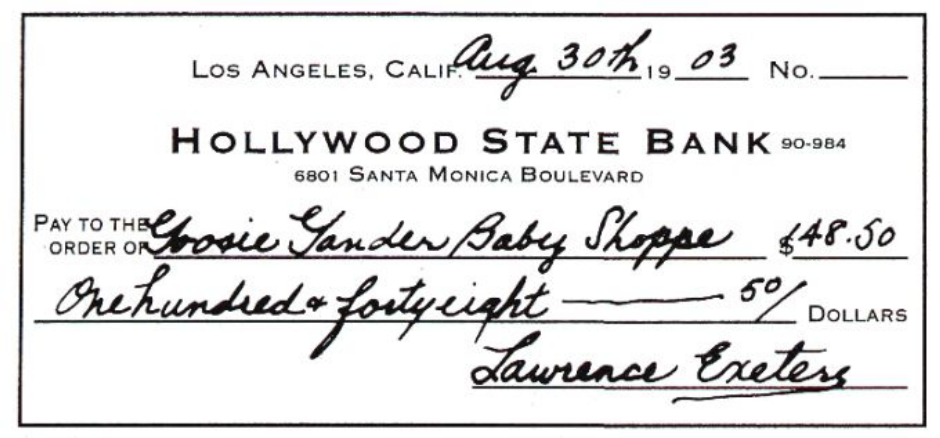
# C. Graph Transformer Networks

- Designed much more easily and quickly by viewing the system as a networks of modules.
  - One or several graphs as input
  - Produce graphs as output
  - Communicate - form of directed graphs



# MULTIPLE OBJECT RECOGNITION: HEURISTIC OVER-SEGMENTATION

- Most recognizers can only deal with one character at a time
- How  GTN used ?
- Avoids the expensive and unreliable task of hand-truthing



Los Angeles, Calif. Aug 30th 19 03 No. ____

Hollywood State Bank 90-984

6801 Santa Monica Boulevard

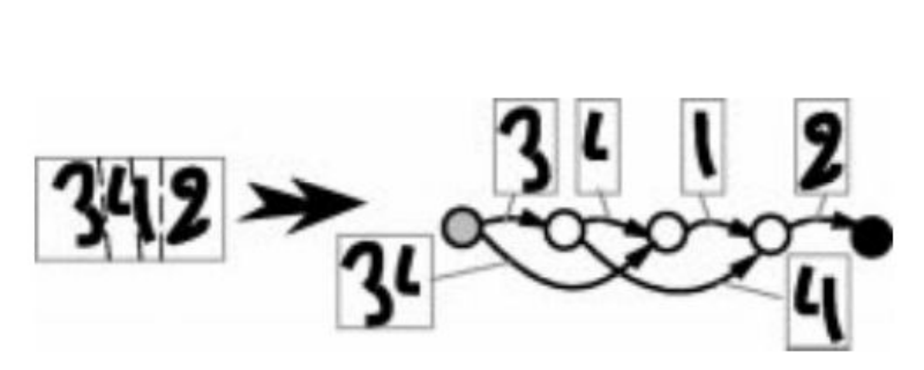
Pay to the order of Goosie Gander Baby Shoppe $48.50

One hundred & forty eight —— 50/ Dollars

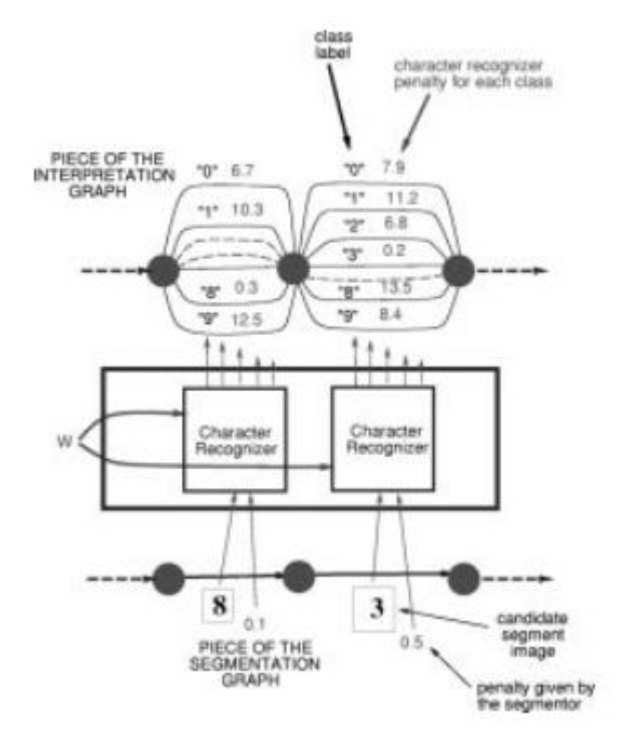
Lawrence Exeter

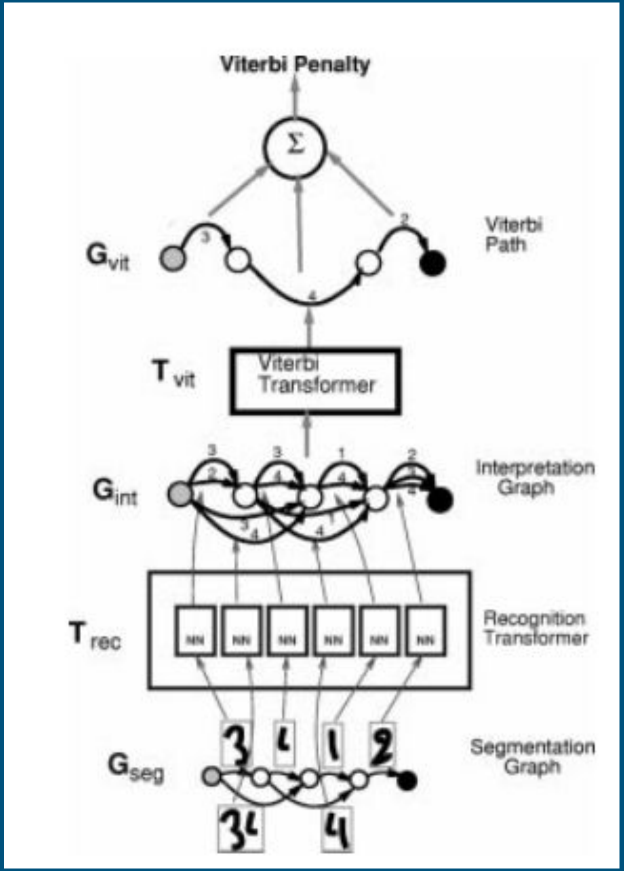
# A. Segmentation Graph

- Use heuristic image processing techniques
- Directed acyclic graph with a start node and an end node



# B.Recognition Transformer and Viterbi Transformer





# GLOBAL TRAINING FOR GRAPH TRANSFORMER NETWORKS

A. Viterbi Training

B. Discriminative Viterbi Training

C. Forward Scoring and Forward Training

D. Discriminative Forward Training

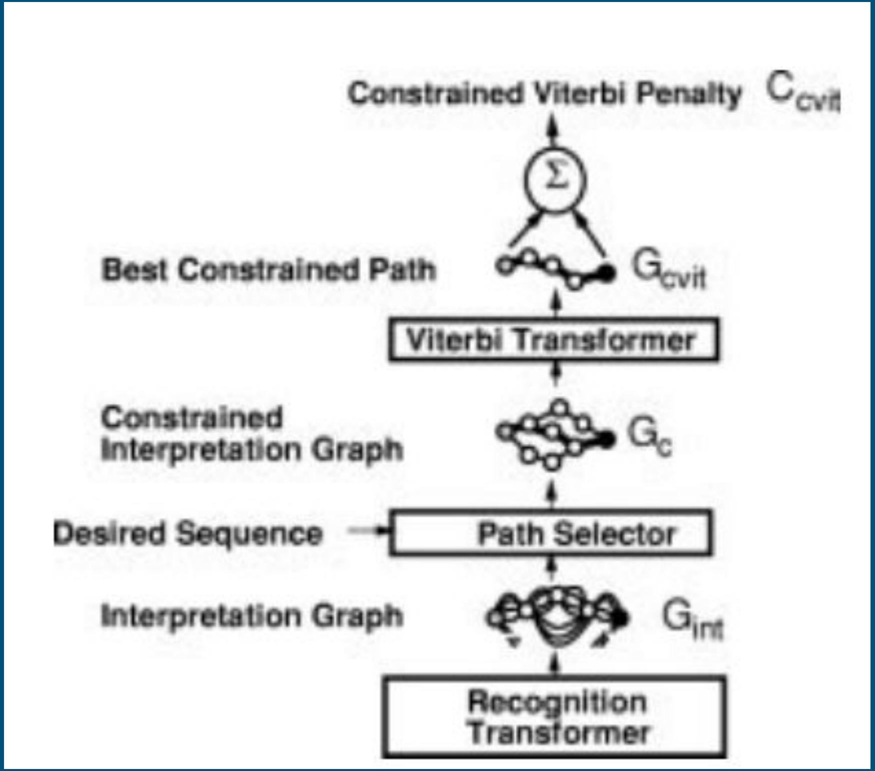
E. Remarks On Discriminative Training

# Viterbi Training

- minimize the average penalty of this “correct” lowest penalty path
- Output is the loss function to current pattern

$$E_{\text{vit}} = C_{\text{cvit}}.$$

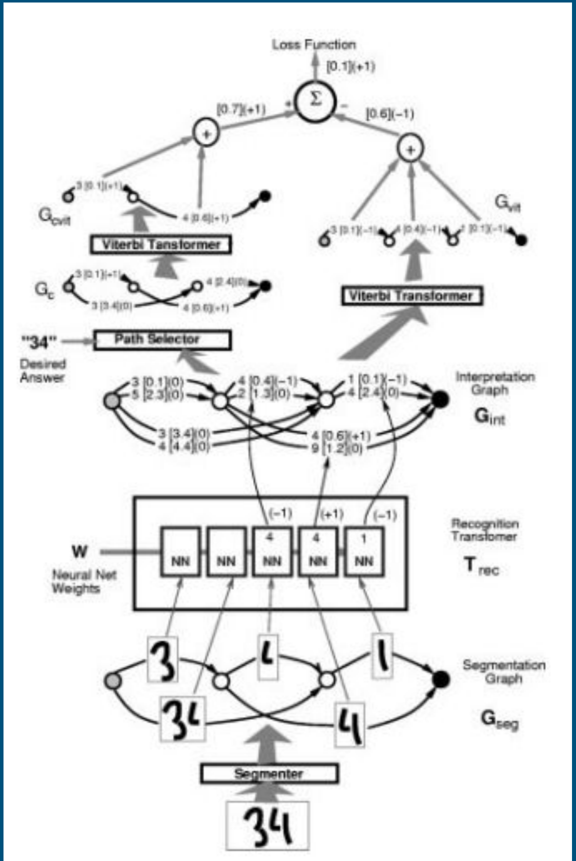
- Loss function is sum of penalties



# Discriminative Viterbi Training

Examples:

- Difference between the penalty of the best correct path and the penalty of the best path
- In error correction procedure
- Used for speech recognition

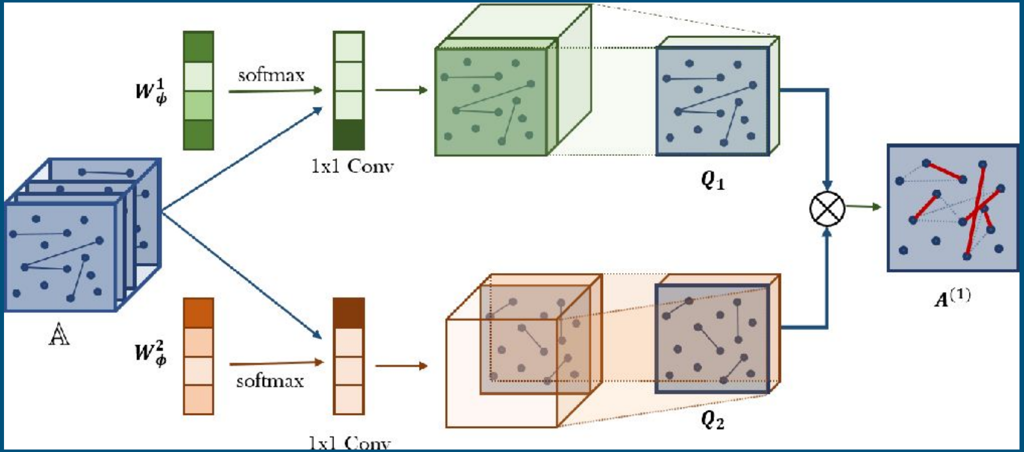


# MULTIPLE OBJECT RECOGNITION: SPACE DISPLACEMENT NEURAL NETWORK

A.Interpreting The Output Of An SDNN With A GTN

B.Experiments With SDNN

# GRAPH TRANSFORMER NETWORKS AND TRANSDUCERS (GTN)



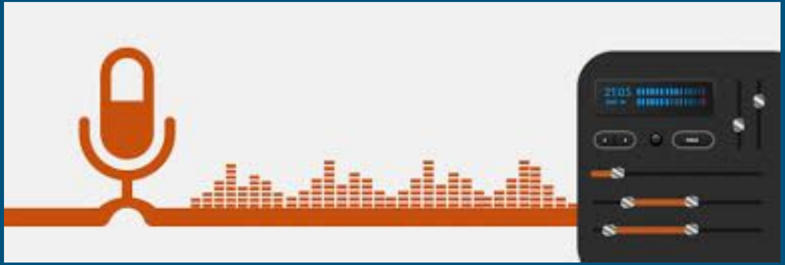
# A. Previous work

- Handwriting Recognition
- Speech Recognition

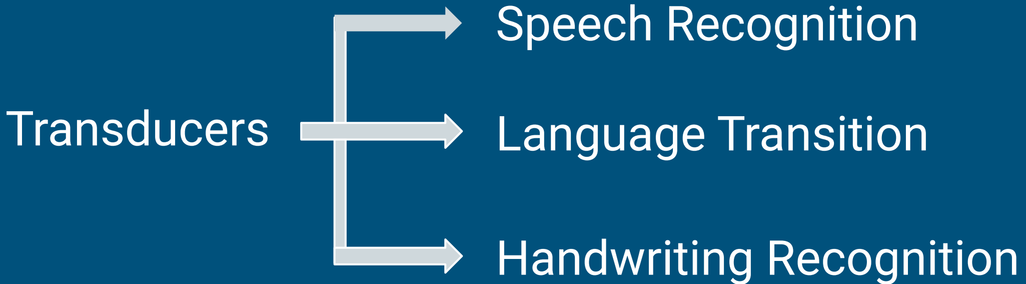
Use Graped-Based learning methods
integrate Graph based statistical models
with acoustic recognition modules mainly
Gaussian mixture models

❏ Problem

No proposal for a systematic approach to multi-layer graph-based Trainable systems.



- Concept of weighted finite-state transducers



Transducers

Speech Recognition

Language Transition

Handwriting Recognition

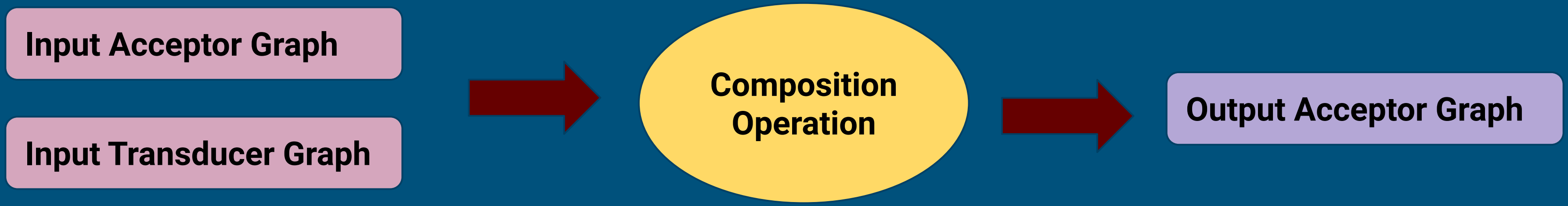
❑ Mainly focused on

>> Efficient search algorithms

>> Algebraic aspects of combining transducers and graphs

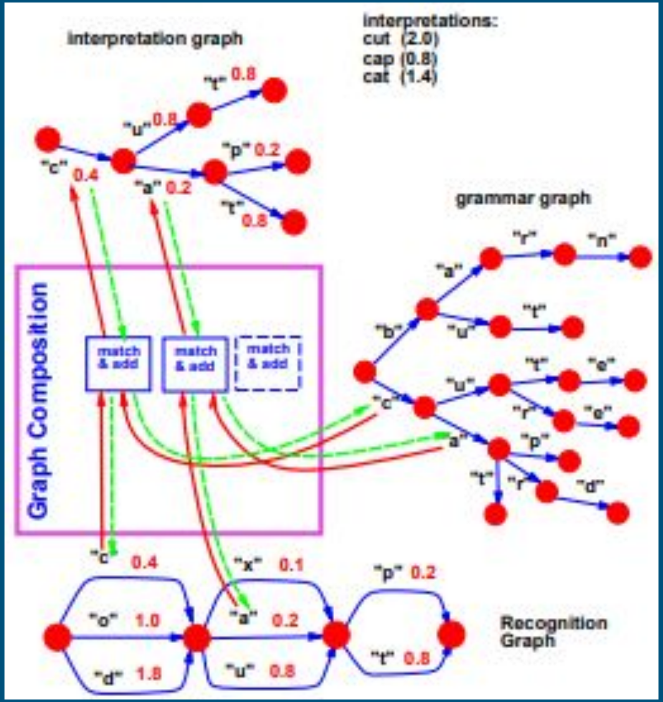
# B. Standard Transduction

- **Transduction Operation**



➢ Each path in this output graph ($S_{out}$) corresponds to one path ($S_{in}$) in the input acceptor graph and one path and a corresponding pair of input/output sequences ($S_{out}$,$S_{in}$) in the transducer graph.

# Composition of the Recognition Graph with the Grammar Graph



- Two tokens sitting each on the start nodes of the input acceptor graph and the transducer graph.
- Trajectory represents a sequence of input symbols that complies with both the acceptor and the transducer.

Eg:-

The incorporation of linguistic constraints when recognizing words or other character strings.

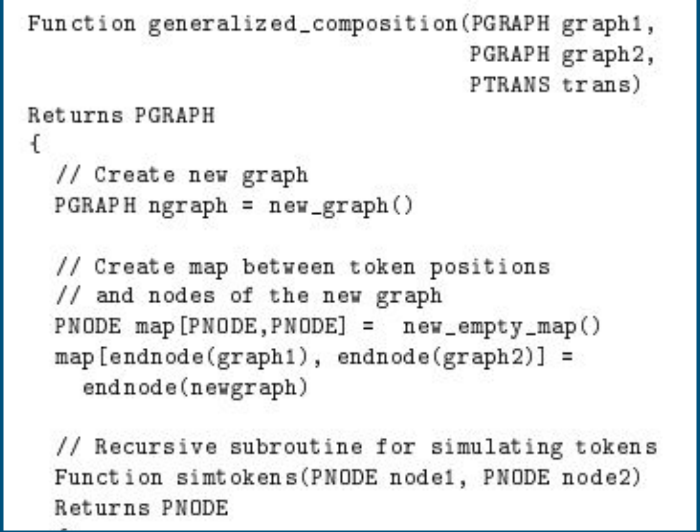
# C. Generalized Transduction

- **Composition Transformer**

Three methods :

1. check(arc1, arc2)
    - Compares the data structures pointed to by arcs arc1 and arc2
    - Returns a boolean
2. fprop(ngraph, upnode, downnode, arc1, arc2)
    - Called when check(arc1, arc2) returns true.
    - Creates new arcs and nodes between nodes upnode and downnode
    - Computes the information attached to these newly created arcs
3. bprop(ngraph, upnode, downnode, arc1, arc2)
    - Called during training
    - Used in the fprop call with the same arguments
    - Compute the values attached to its output arcs is differentiable

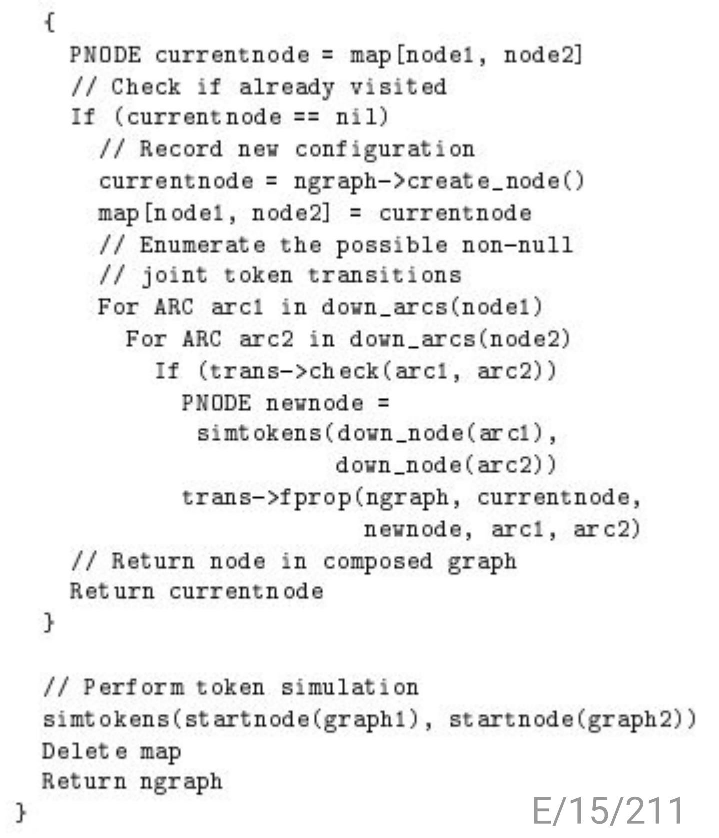
# Simplified generalized graph composition algorithm



```
Function generalized_composition(PGRAPH graph1,
                                 PGRAPH graph2,
                                 PTRANS trans)
Returns PGRAPH
{
  // Create new graph
  PGRAPH ngraph = new_graph()

  // Create map between token positions
  // and nodes of the new graph
  PNODE map[PNODE,PNODE] =  new_empty_map()
  map[endnode(graph1), endnode(graph2)] =
    endnode(newgraph)

  // Recursive subroutine for simulating tokens
  Function simtokens(PNODE node1, PNODE node2)
  Returns PNODE
```



```
  {
    PNODE currentnode = map[node1, node2]
    // Check if already visited
    If (currentnode == nil)
      // Record new configuration
      currentnode = ngraph->create_node()
      map[node1, node2] = currentnode
      // Enumerate the possible non-null
      // joint token transitions
      For ARC arc1 in down_arcs(node1)
        For ARC arc2 in down_arcs(node2)
          If (trans->check(arc1, arc2))
            PNODE newnode =
             simtokens(down_node(arc1),
                       down_node(arc2))
            trans->fprop(ngraph, currentnode,
                         newnode, arc1, arc2)
    // Return node in composed graph
    Return currentnode
  }

  // Perform token simulation
  simtokens(startnode(graph1), startnode(graph2))
  Delete map
  Return ngraph
}
```

E/15/211

# D. Notes On The Graph Structure

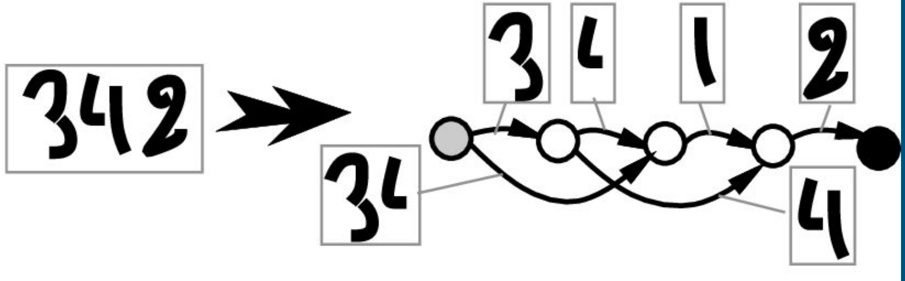
| bprop( ) | check( ) | fprop( ) |
|---|---|---|
| Basis of the backpropagation algorithm for generic graph transformers | Establishes the structure of the ephemeral network inside the composition transformer | Implements the numerical relationship |

➢ The structure of the graph depends on
  - the nature of the Graph Transformer
  - the value of the parameters and on the input

This might be considered a combinatorial problem and not amenable to Gradient-Based Learning

# E. GTN And Hidden Marcov Model

- **GTN**



- ➢ A generalization and an extension of HMMs.
- ➢ Extend HMMs by allowing to combine in a well-principled framework multiple levels of processing, or multiple models

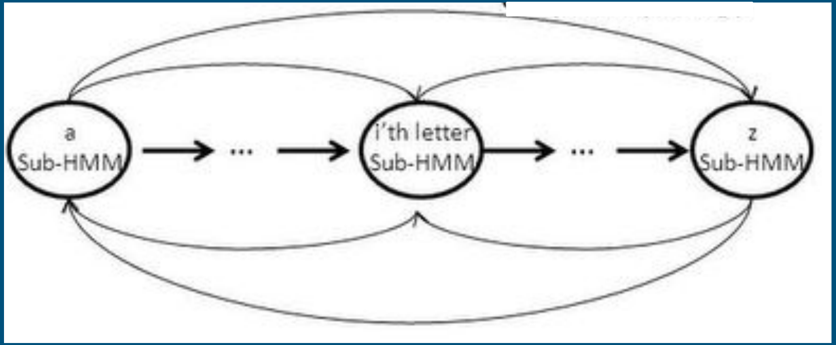
Eg:- Pereira et al. have been using the transducer framework for stacking HMMs

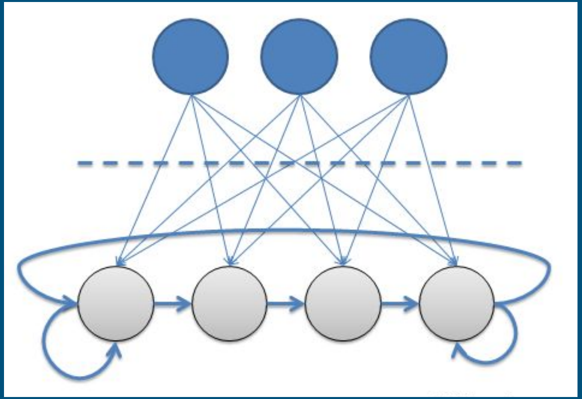
## ● Hidden Marcov Model (HMM)

➢ It has nodes n(t , i) associated to each time step t and state i in the model

➢ The Input-Output HMM (IOHMM) model is strongly related to graph transformers

➢ IOHMM represents the conditional distribution of output sequences given input sequences

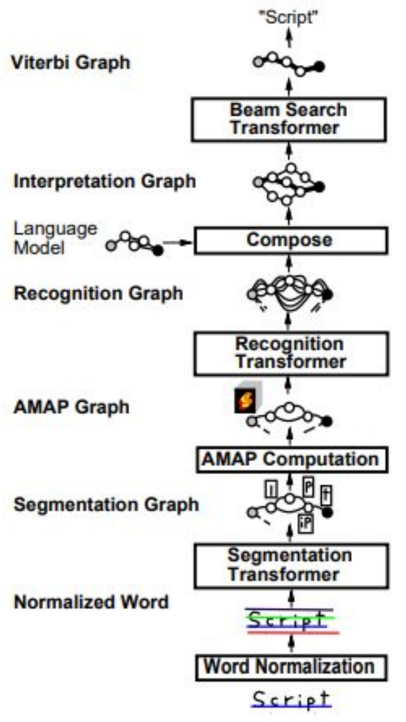




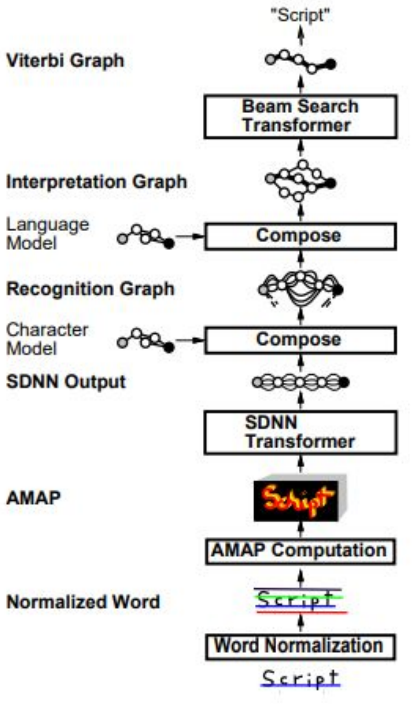
# An On-Line Handwritten Recognition System

- Natural handwriting is a mixture of different “styles"
- A  recognizer is required to improve interaction with pen-based devices to identify handwritings.

- Word recognition system for pen-based devices based on,
  1. Preprocess
  2. Module
  3. Replicated convolutional neural network
  4. GTN

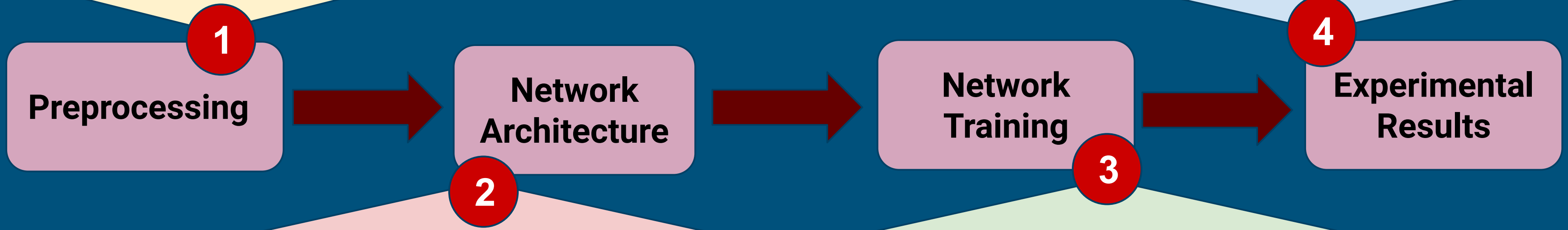


An online handwriting recognition GTN based on heuristic over- segmentation



An online handwriting recognition GTN based on Space Displacement Neural Network

# System Flow



1

- Reduces intra-character variability, simplifying character recognition.
- Design AMAP where pen trajectories are represented by low resolution images

2

- Best online/offline network is a 5-layer convolutional network
- Determine that the resolution was sufficient for representing handwritten characters

3

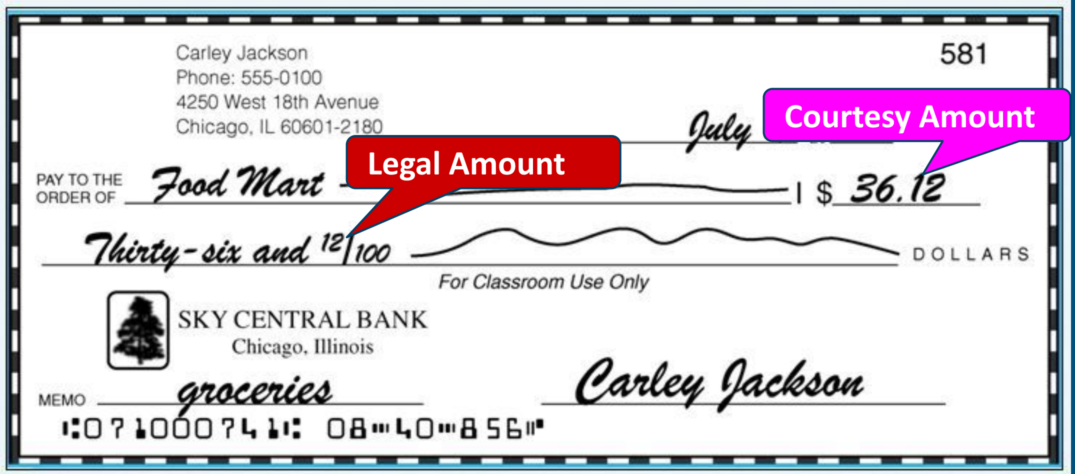
- Training proceeded in two phases
  **1.** Kept the centers of the RBFs fixed
  **2.** All the parameters, network weights, RBF centers were trained globally
- Two approaches
  1. Heuristic Over-Segmentation approach
  2. SDNN approach

4

- Three set of experiments
  1. Evaluate the generalization ability of the neural network classifier
  2. Obtain with character-level normalization & word and character errors
  3. Obtain with the joint training of the neural network & the post-processor with the word-level criterion

# A Check Reading System

- Task
  - Verification of the amount on a check
  - Read the Courtesy amount only.



Carley Jackson
Phone: 555-0100
4250 West 18th Avenue
Chicago, IL 60601-2180

581

July

PAY TO THE ORDER OF Food Mart | $ 36.12

Thirty-six and 12/100 DOLLARS

For Classroom Use Only

SKY CENTRAL BANK
Chicago, Illinois

MEMO groceries

Carley Jackson

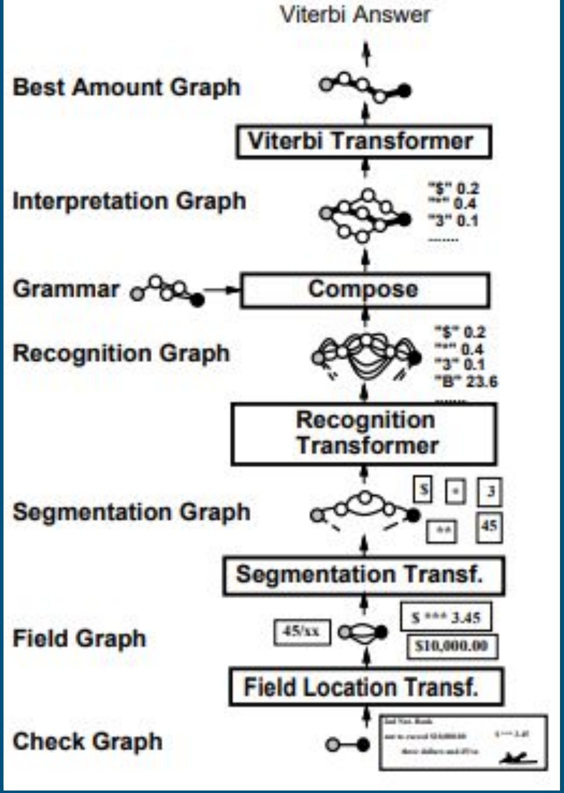
⑆071000741⑆ 08⑈40⑈856⑈

Legal Amount
Courtesy Amount

- Two main steps

1. Find the candidates that are the most likely to contain the courtesy amount among all fields
2. Find the best interpretation of the amount using contextual knowledge represented by a stochastic grammar for check amounts

# A. A GTN for Check Amount Recognition



**Procedure :**

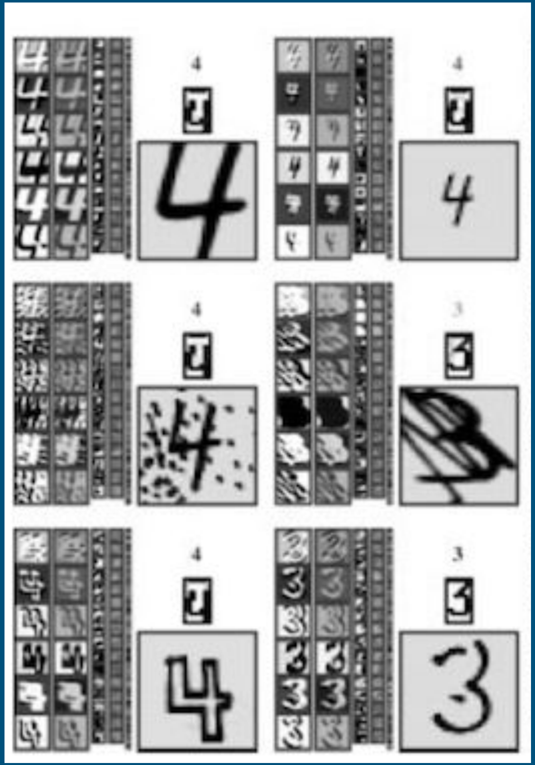
Input - A trivial graph with a single arc that carries the image of the whole check

- $T_{field}$ - The field location transformer
- $T_{seg}$ - The segmentation transformer
- $T_{rec}$ - The recognition transformer
- $T_{gram}$ - The composition transformer
- The Viterbi Transformer

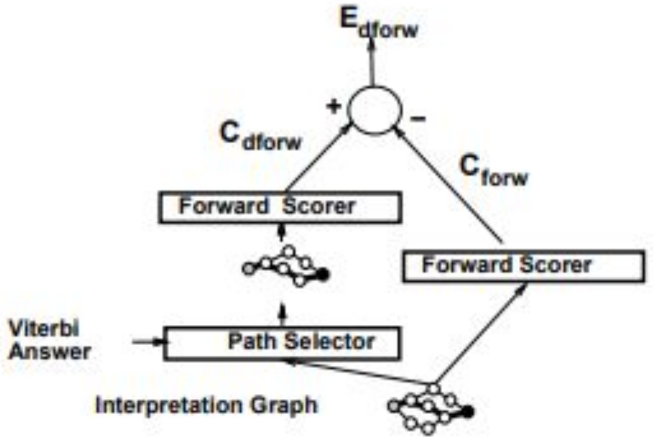
# B. Gradient-Based Learning

- The parameters of the field locator and the segmenter are initialized by hand, while the parameters of the neural network character recognizer are initialized by training on a database of pre-segmented and labeled characters

- Prior to globally optimizing the system, each module parameters are initialized with reasonable values.



# C. Rejecting Law Confidence Checks



$$\boxed{\text{Confidence} = \exp(E_{dforw})}$$

# D. Results

Number of character images used to train = 500,000

Number of business checks used to measure the performance = 646

| | Machine-printed performance | Handwritten performance |
|---|---|---|
| **Correctly recognized** | 82% | 68% |
| **Errors** | 1% | 1% |
| **Rejects** | 17% | 31% |

# Conclusions

- In neural networks, backpropagation allows us to efficiently compute the gradients on the connections  of the neural network, with respect to a loss function.
- Hand crafted should be replaced by an automatic learned features.
- Larged sized systems can be learned by gradient based methods with efficient back propagation

# Thank You!!!!