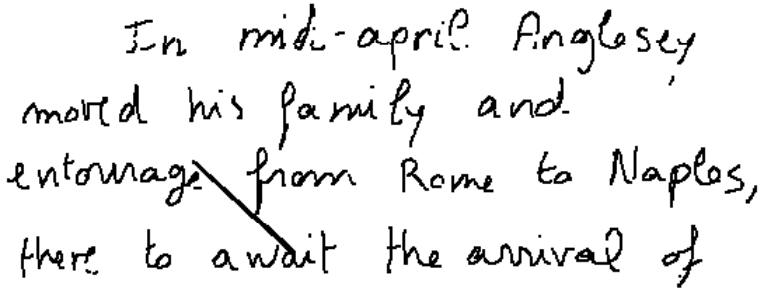
Réseaux de neurones récurrents
Handwriting Recognition with Long Short-Term Memory Networks

Dr. Marcus Eichenberger-Liwicki

DFKI, Germany

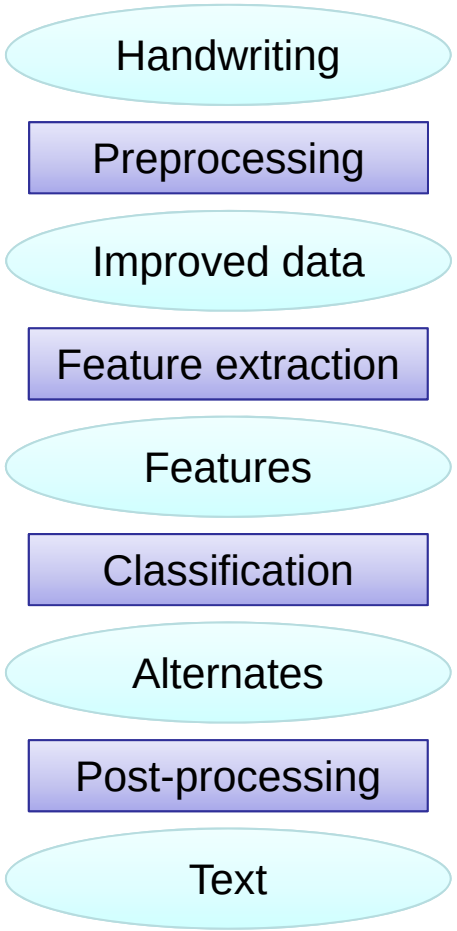
Marcus.Liwicki@dfki.de

- Transform handwritten text into a machine readable format



In mid-april Anglesey
moved his family and
entourage from Rome to Naples,
there to await the arrival of

Handwriting Recognition Sub-Tasks



- Pixel data or (x,y)-sequence
- Enhance signal
- Transform data into real numbers
- Apply ANN, kNN, GMM, HMM, …
- Or BLSTM / MDLSTM
- Use language information

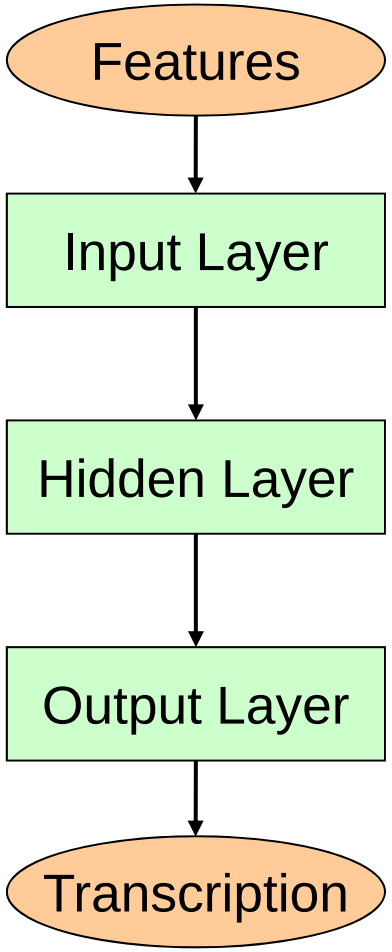
Short Introduction of myself

- 2001-2004 Free University of Berlin – Master of Computer Science
- 2004-2007 PhD study in Bern – Dr. phil. nat. and PhD
- 2008- Senior Researcher and lecturer at DFKI
- 2009-2010 JSPS research fellow at Kyushu University (Seiichi Uchida)

- More than 60 publications including one book, one book chapter and 10 journal papers

The Novel Classifier

- Input: raw features, raw pixel data, or raw point-sequence
- Output machine-readable transcription
- Easy to use
- Based on Multi-Layer Perceptron



- Feature vector at timestamp t

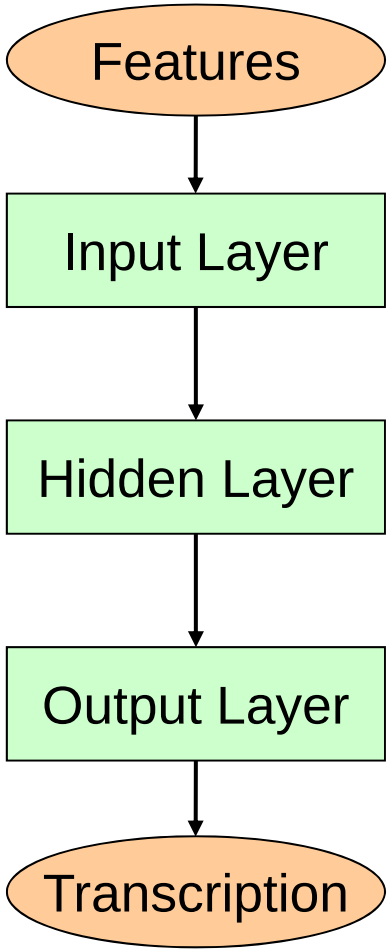
$$x_1^t, \ldots, x_n^t$$

- Perceptrons in the individual layers
  - Aggregation function $a^t = \sum w_i x_i^t$
  - Activation function (squashing f.)

$$b_h^t = h(a^t)$$

$$h: \quad \tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

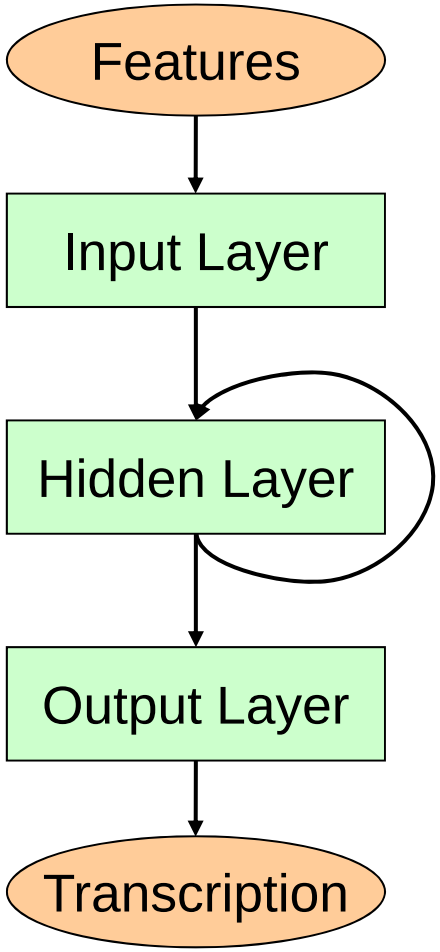
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



- For every hidden layer:

$$b_h^t = h(\sum w_{h'h} b_{h'}^t)$$

- Neuron with highest output corresponds to the recognized class
- Training with backpropagation



- Recurrent connections are added in order to keep information of previous time stamps in the network
- Novel equation for activation:

$$a^t = \sum w_i x_i^t + \sum w_h b_h^{t-1}$$

- Can be written in matrix form

$$A^t = W_i \cdot X^t + W_h \cdot B^{t-1}$$

- Context information is used, however

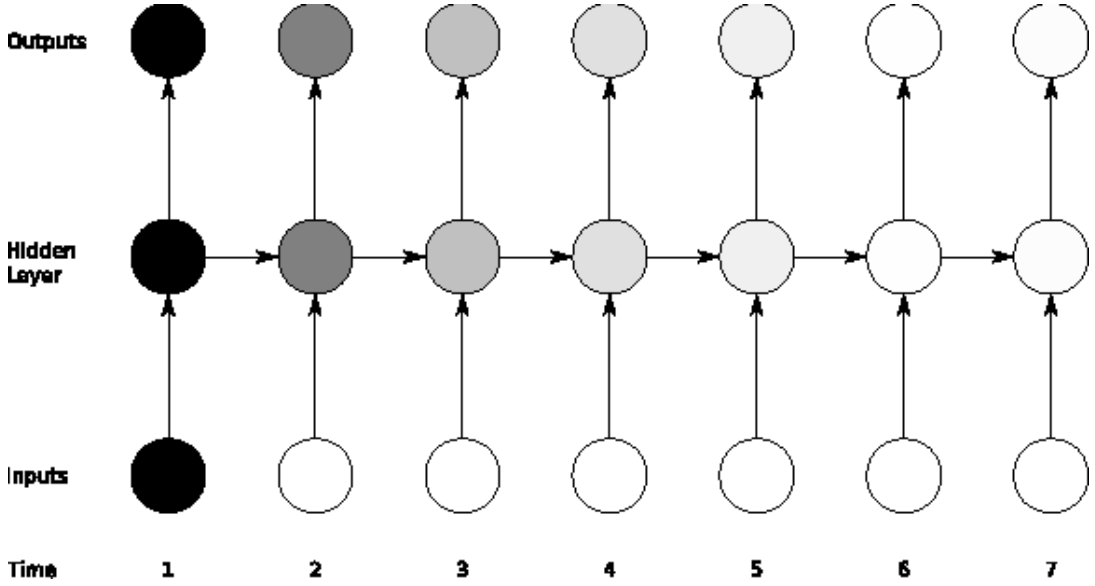
- Usual RNN forget information after a short period of time

$$A^t = W_i \cdot X^t + W_h \cdot B^{t-1} = W_i \cdot X^t + W_h \cdot h(W_i \cdot X^{t-1} + W_h \cdot B^{t-2})$$

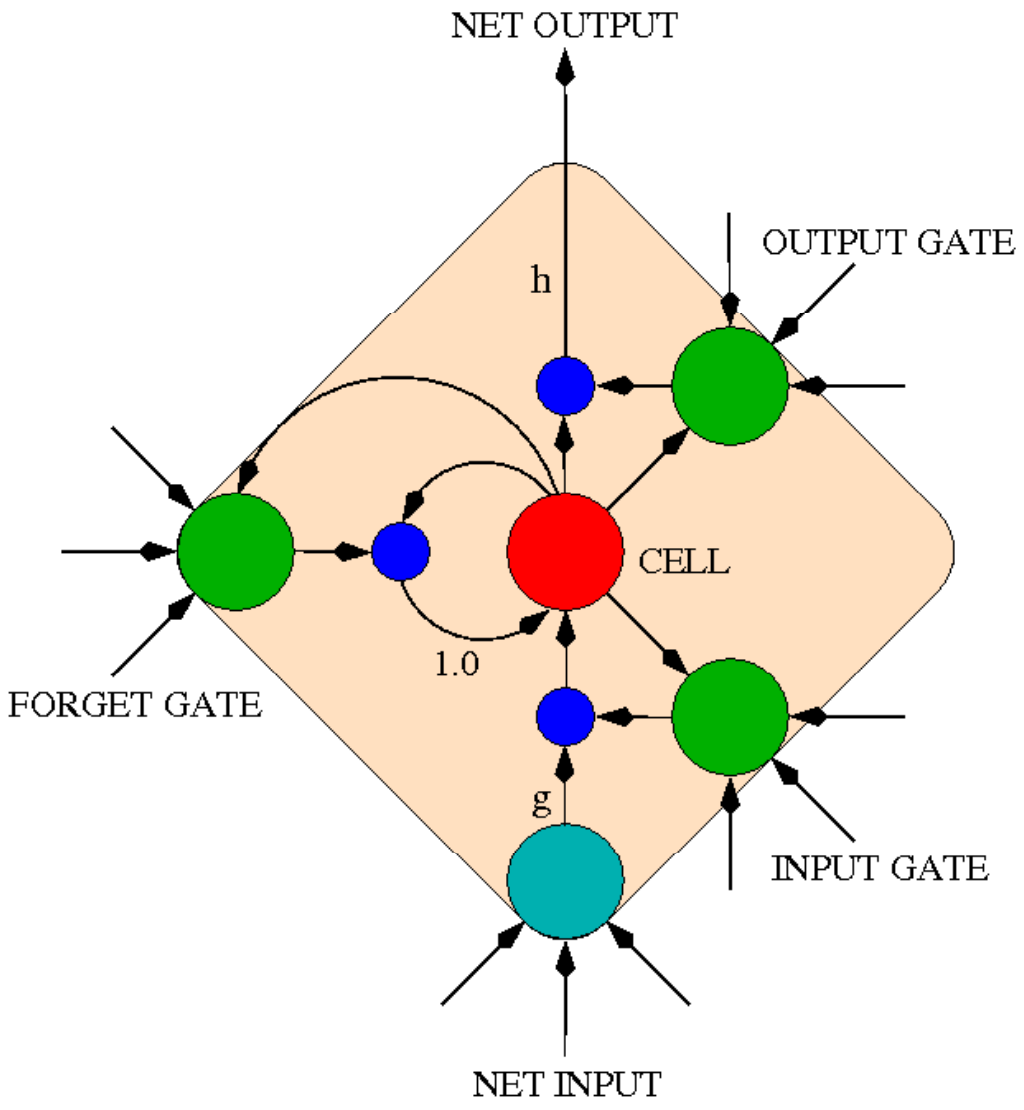
$$\Rightarrow A^t = f(X^t, W_h X^{t-1}, {W_h}^2 X^{t-2}, \ldots, {W_h}^t X^1)$$

- Example:
  Neuron
  During
  7 timestamps

- Information
  vanished



Core Idea: New Memory Cell Instead of Perceptron



- Memory cell
  - Read, write and reset operations
- Input Gate (single cell)

$$a_\iota^t = W_{a,i} \cdot X^t + W_{h,i} \cdot B^{t-1} + W_{c,i} S_c^{t-1}$$

- Forget Gate

$$a_\theta^t = W_{h,\theta} \cdot X^t + W_{h,\theta} \cdot B^{t-1} + W_{c,\theta} S_c^{t-1}$$

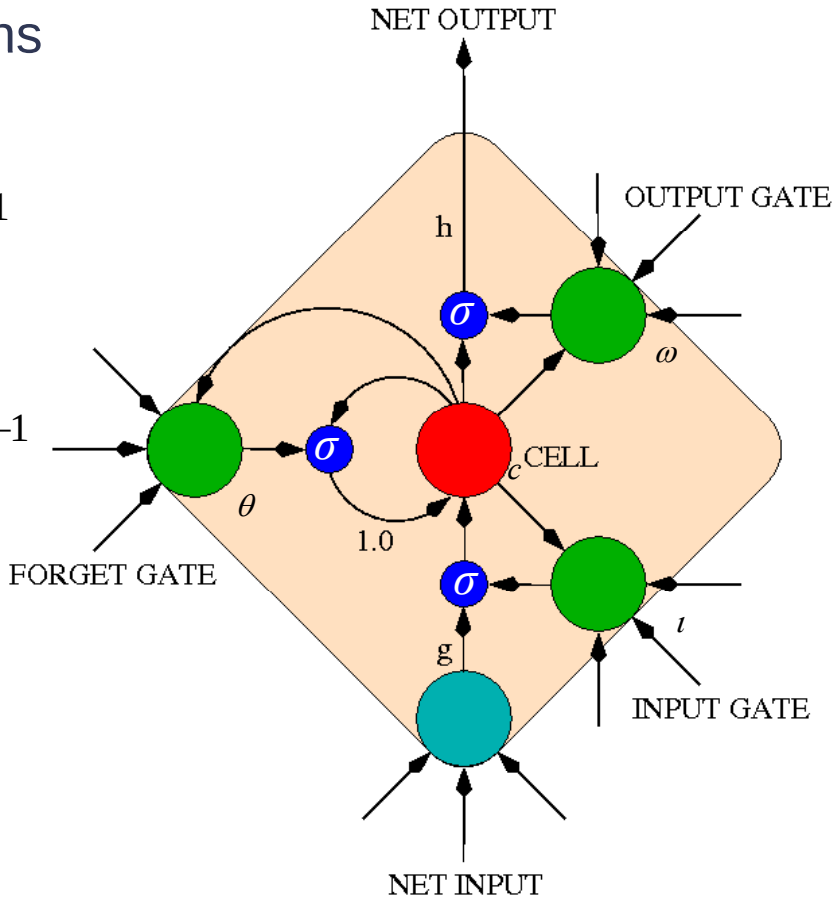
- Cell State

$$a_c^t = W_{a,c} \cdot X^t + W_{h,c} \cdot B^{t-1}$$

$$s_c^t = \sigma(a_i^t) g(a_c^t) + \sigma(a_\theta^t) s_c^{t-1}$$

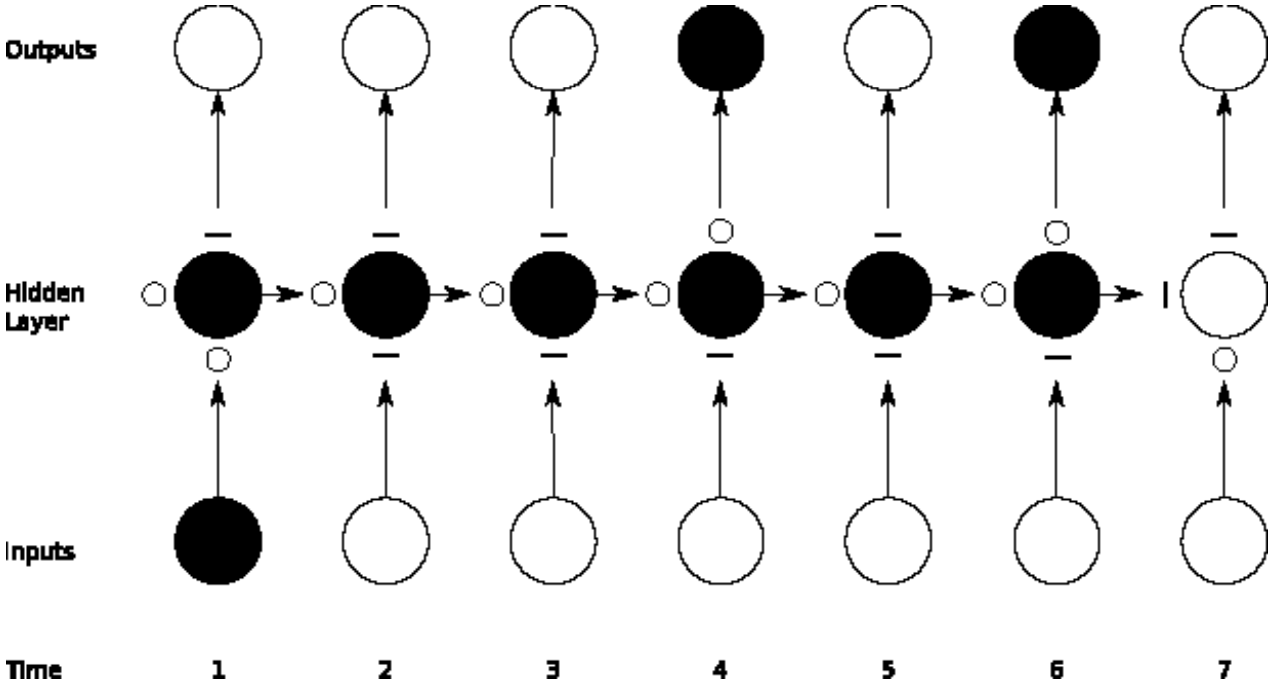
- Assume σ is close to 0 or 1

$$\Rightarrow s_c^t = [0 or 1] g(a_c^t) + [0 or 1] s_c^{t-1}$$



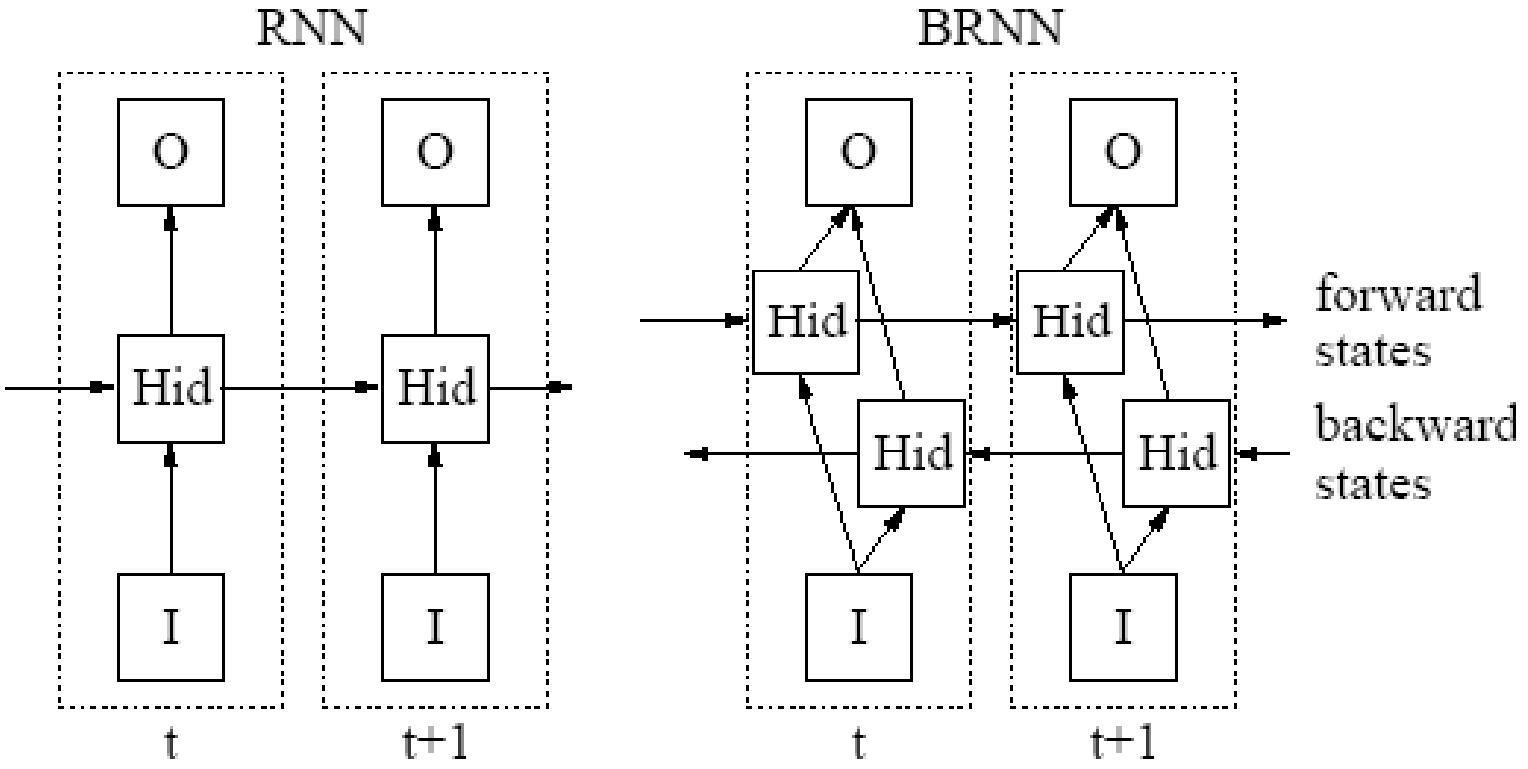
No Vanishing Gradient

- Output Gate $a_\omega^t = W_{a,\omega} \cdot X^t + W_{h,\omega} \cdot B^{t-1} + W_{c,\omega} S_c^t$
- Output $b_c^t = \sigma(a_\omega^t) h(s_c^t)$
- Neuron now
  - O : open ($\sigma=1$)
  - | : closed ($\sigma=0$)



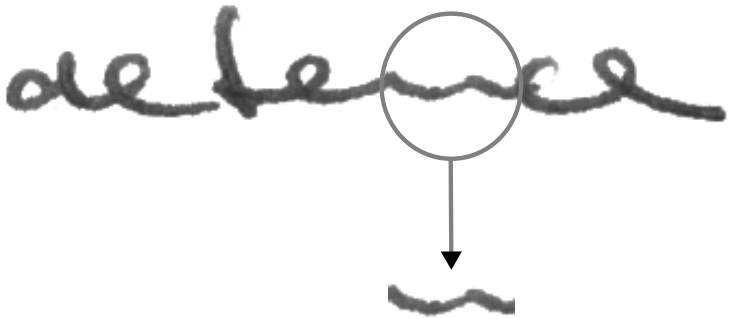
Discussion: MLP - LSTM

Bidirectional RNN

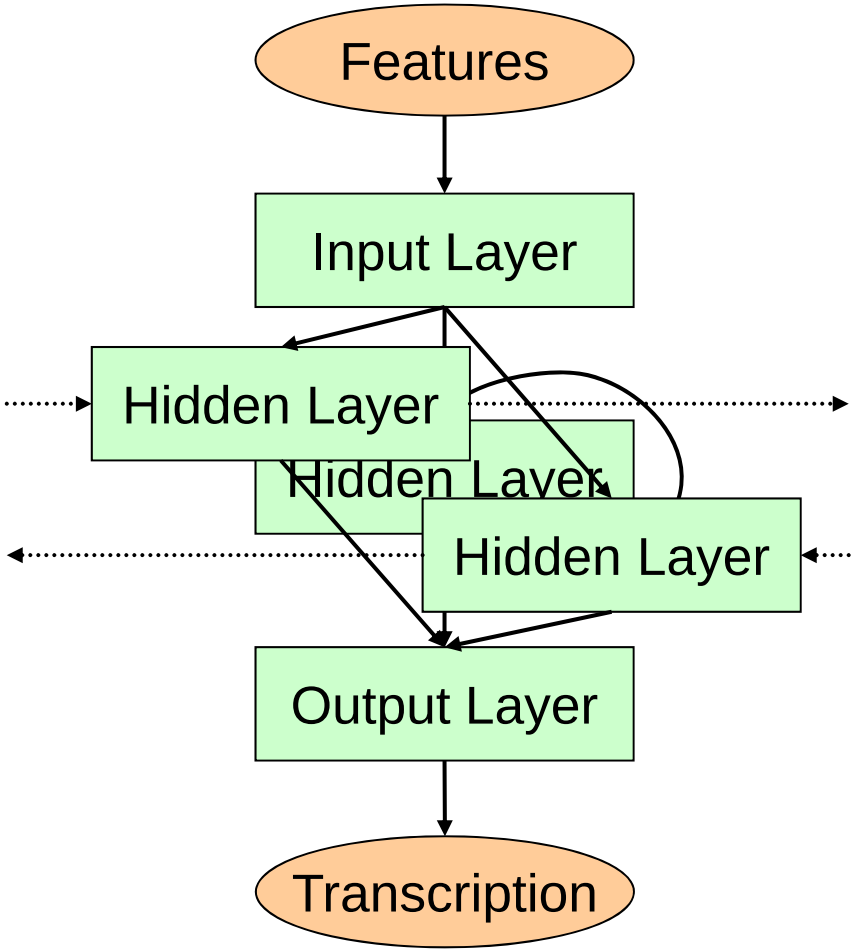


- Trained with backpropagation through time (forward path trough all time stamps for each hidden layer sequentially)

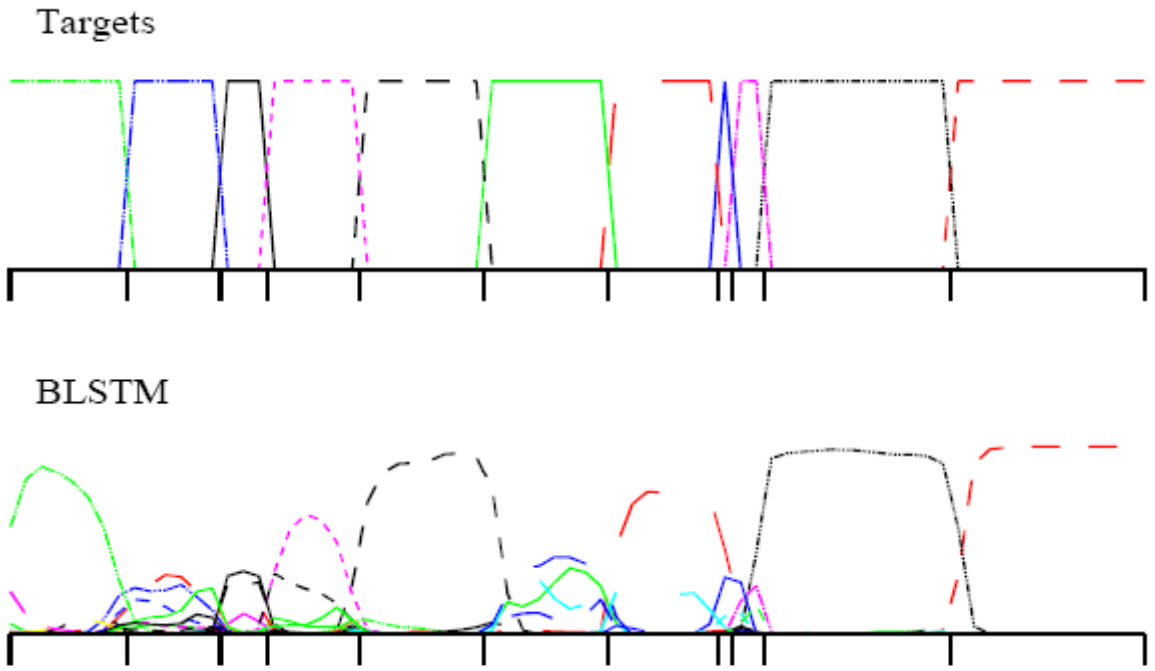
- Importance of context



- Multilayer perceptron network
- Recurrent connections
- Bidirectional
- Memory instead of perceptron

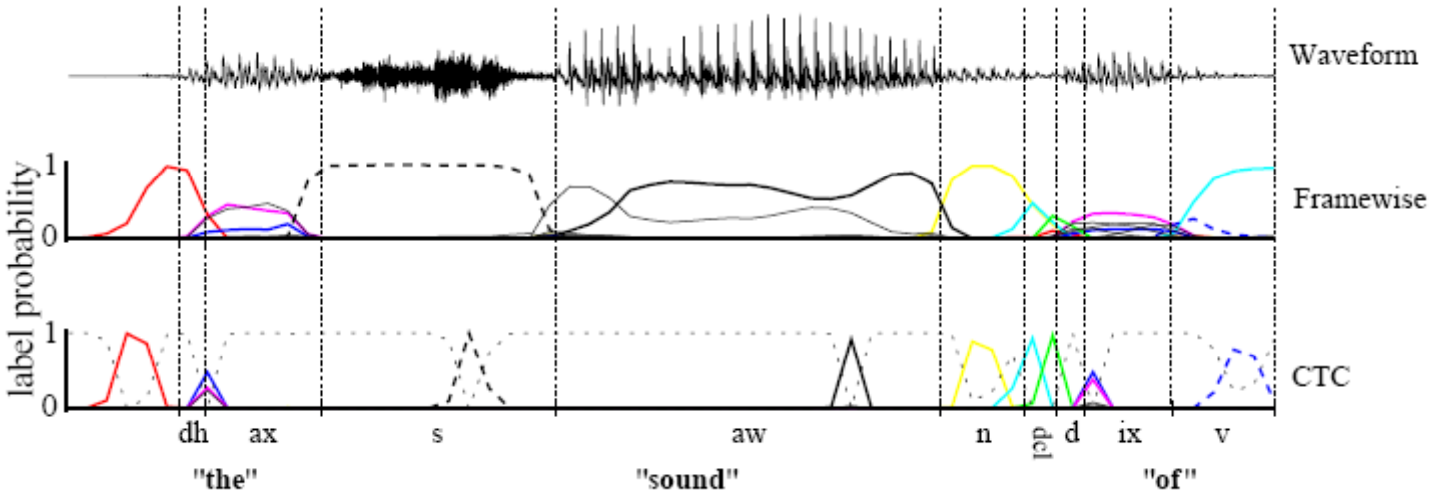


Standard Framewise Classification

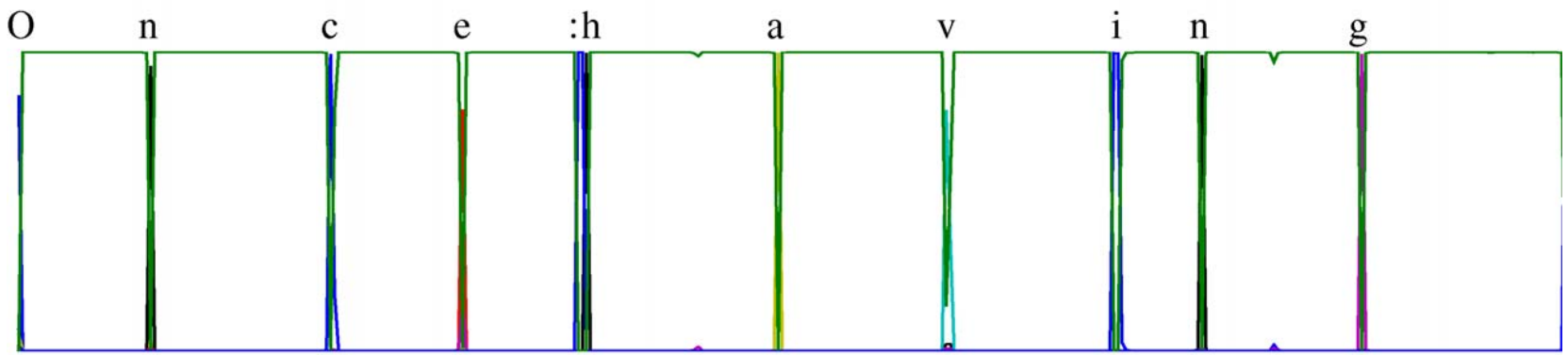


- Segmentation is needed
- Information of previous and next frames is not available
- Idea: introduce a way of connected temporal classification

- Connected Temporal Classification (CTC)
- Example for speech:



- Additional blank label (b green)
- Allows application to whole sequences
- Output with normalized likelihood for each word



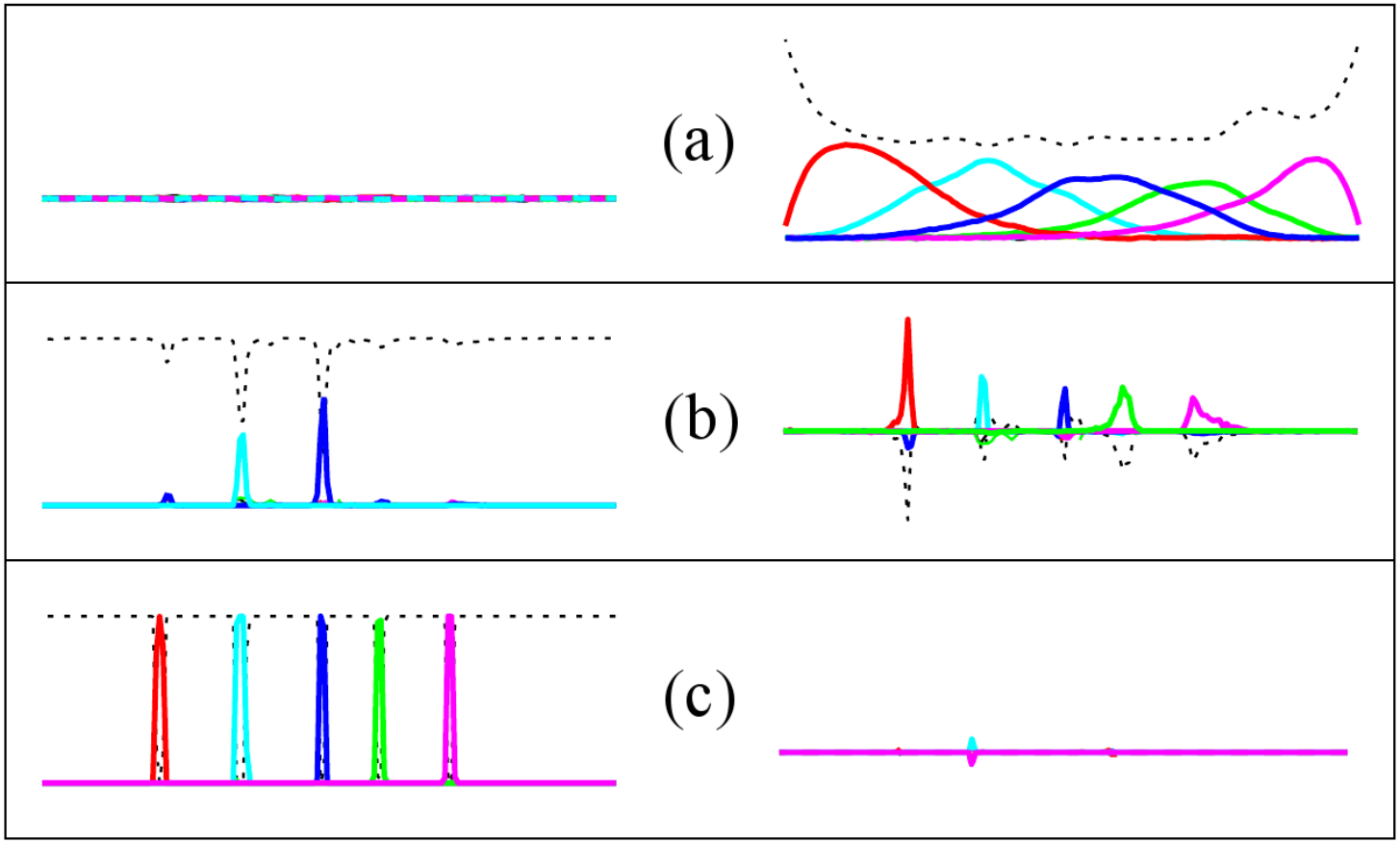
- Training: objective function is smoothed and recalculated after each iteration (details in references)
- Testing: similar to Viterbi-algorithm

How Does it Behave? Training Error Example

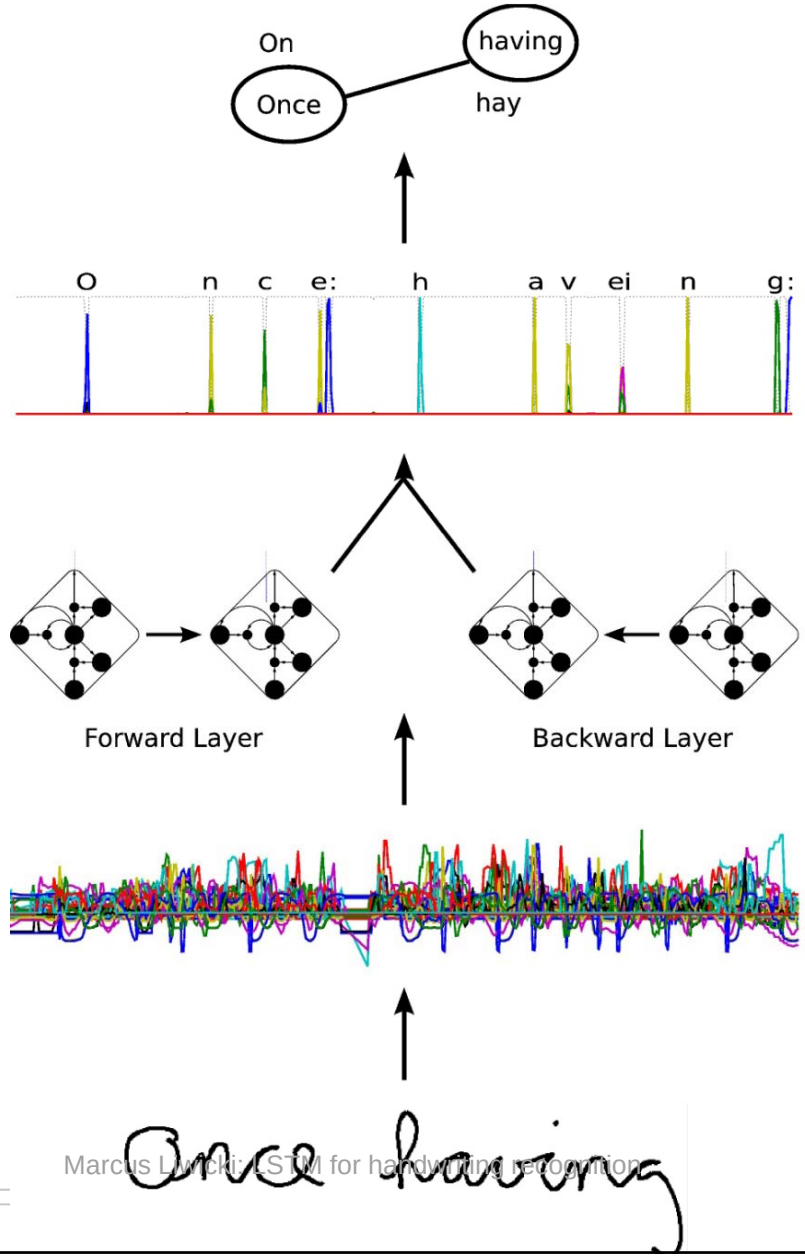
Beginning (random)

10 iterations (error aroud predictions)

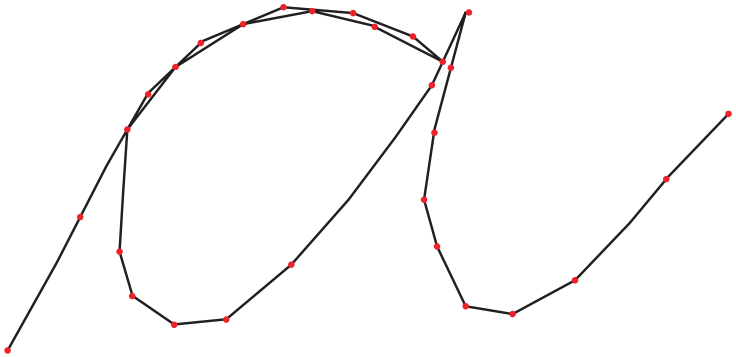
Final (nearly no error)

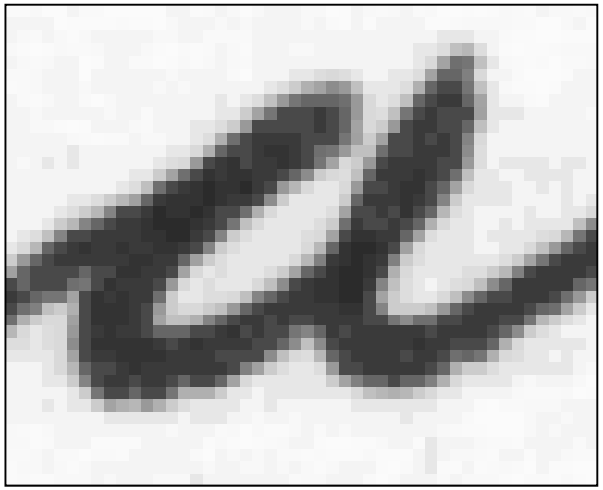


Overall System



- On-line
  - Sequence information
- Off-line
  - No sequence information
  - Grayscale image





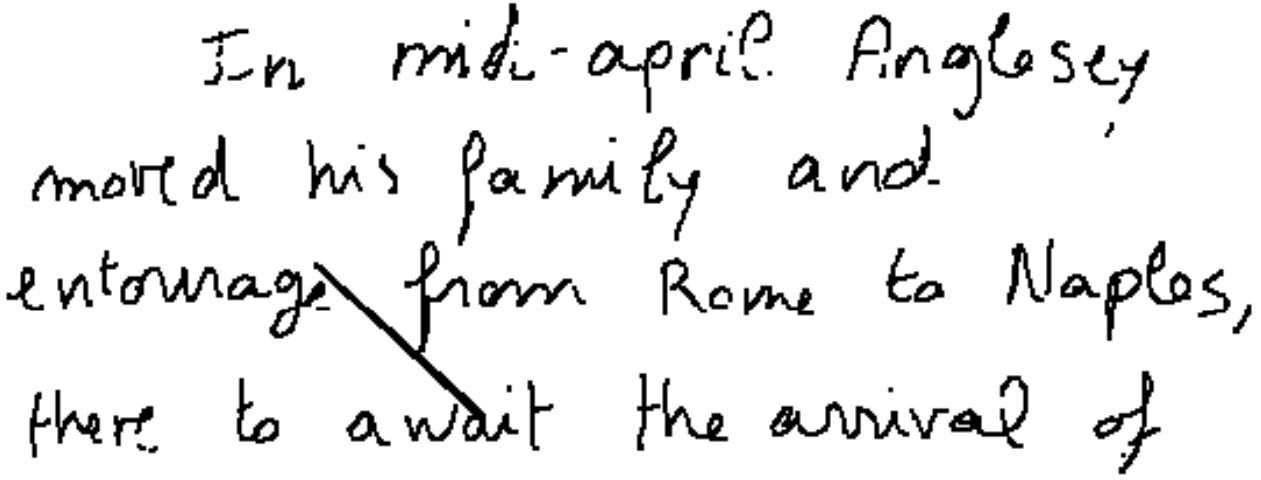
- Spurious points removal

  For each point p_i of a stroke:

  If $d(p_i, p_{i-1}) > \theta$ and $N_\delta(p_i) < N_\delta(p_{i-1})$ then remove p_i ,

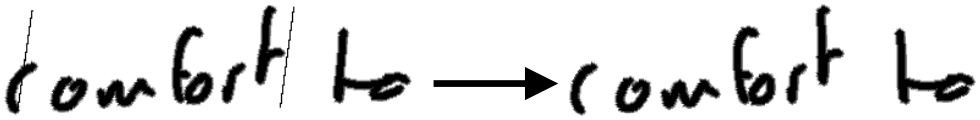
  where $N_\delta(p_i)$ is the number of points from the stroke around p_i

- Text line extraction

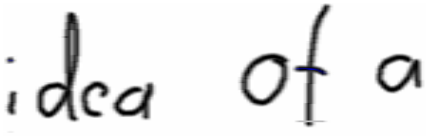


Online Feature Extraction

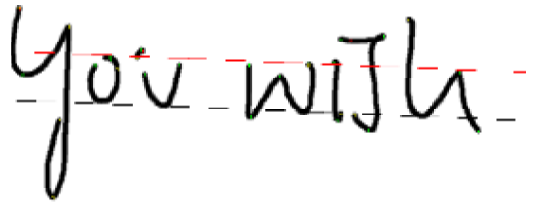
- Skew and slant correction



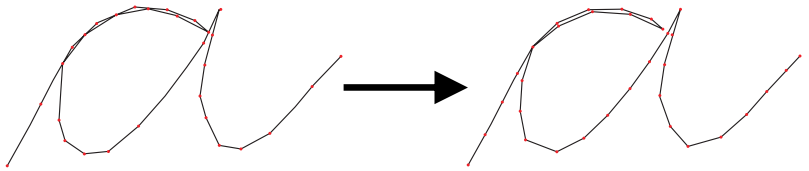
- HAT-removal



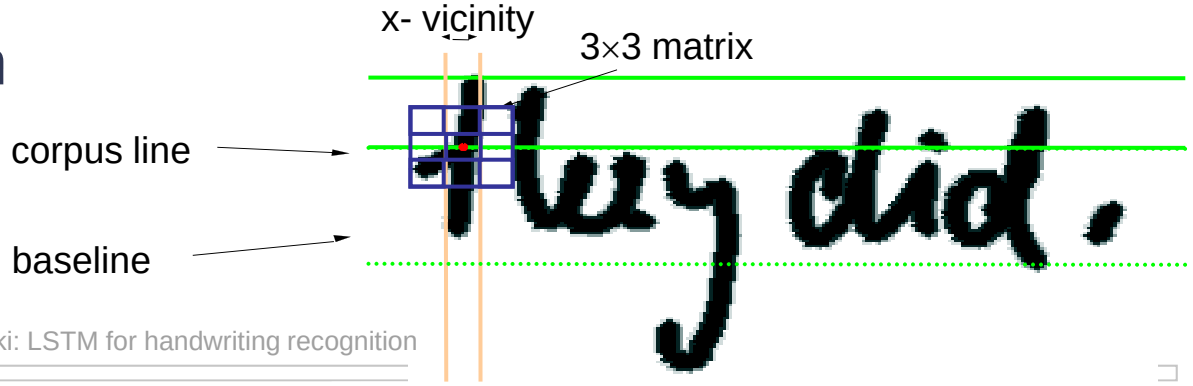
- Size normalization
  - Baseline and corpus line estimation
  - Height normalization
  - Width $N_{b/w} * c$



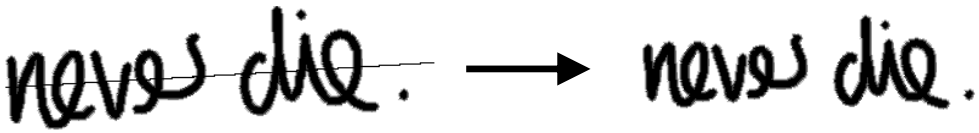
- Equidistant resampling



- Feature extraction
  - Online



- Transformed images
- Skew correction

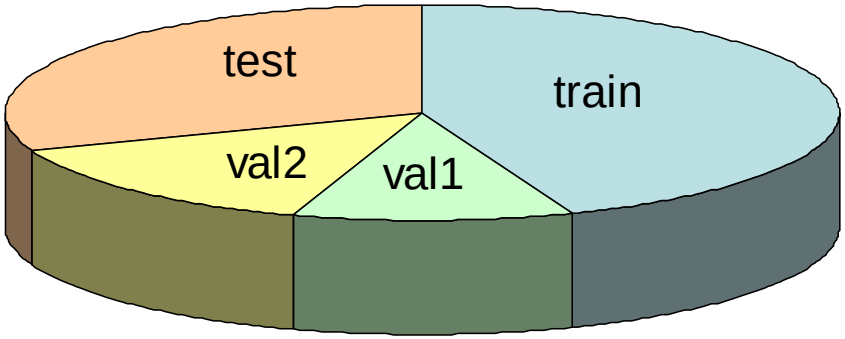


  - Linear regression through the n lowest points $\{p_i = (x_i, y_i)\}$
  - Baseline $y = a * x + b$

$$a = \frac{\mathrm{Cov}(x, y)}{\mathrm{Var}(x)} = \frac{\sum_{i=1}^{n} x_i y_i - n * \overline{xy}}{\sum_{i=1}^{n} x_i^2 - n * \overline{x}^2}, \qquad b = \overline{y} - a * \overline{x}$$

- Slant correction
  - Angle histogram through vertical contour
  - Shearing to upright position
- Feature extraction
  - Nine features using a sliding window approach

- IAM-OnDB-t2 benchmark[1]:
  - Training set, two validation sets, test set
  - Open vocabulary, 82 characters, 5.8% OOVs

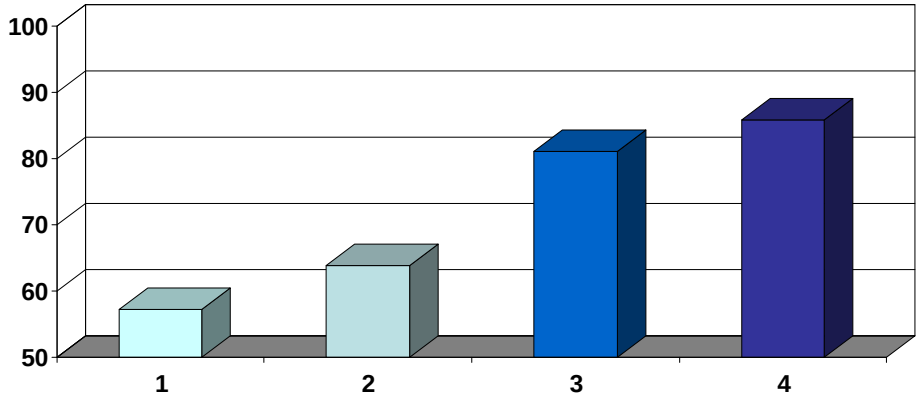


- LM trained on three corpora (Brown, LOB, Wellington)

- Accuracy measured on the word level

Experiments

- Combination
  - Plurality voting
  - Confidence-based voting
- Experiments
  1. 57.34% with off-line (HMM)
  2. 63.86% with on-line
  3. 81.05% with on-line CTC (74.00% off-line)
  4. 86 % after combination of several classifiers

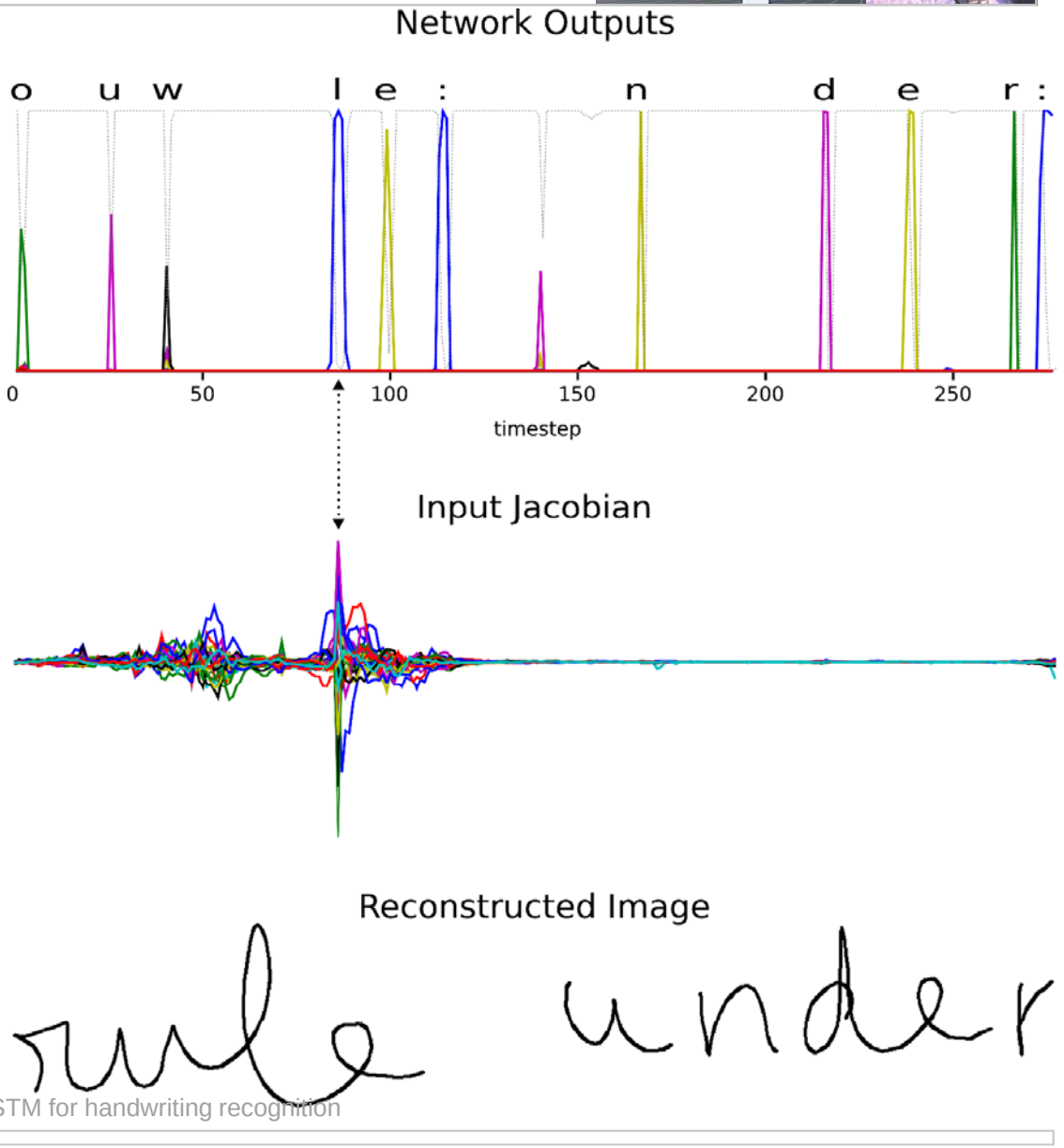


Raw vs. Processed data

System	Input	LM	WER
HMM	Preprocessed	✓	35.5%
CTC	Raw	✗	$30.1 \pm 0.5\%$
CTC	Preprocessed	✗	$26.0 \pm 0.3\%$
CTC	Raw	✓	$22.8 \pm 0.2\%$
CTC	Preprocessed	✓	$20.4 \pm 0.3\%$

- LM – language model

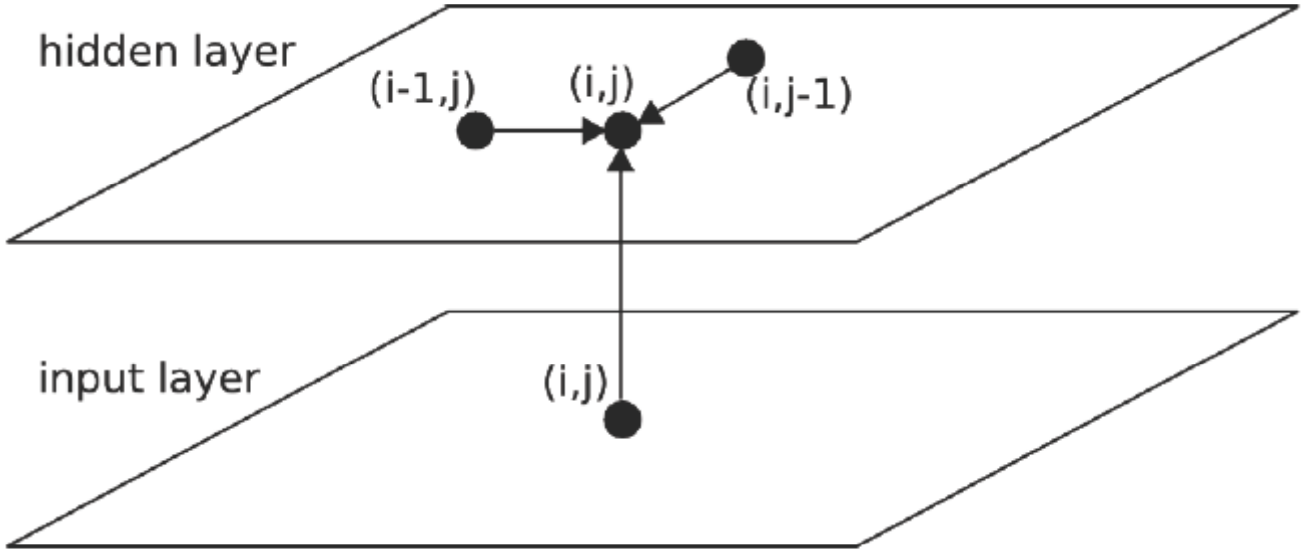
- Example
  - Output at l
  - Amount of information for each cell derived from each time stamp
  - Called input Jacobian
  - Estimated by Backprop.



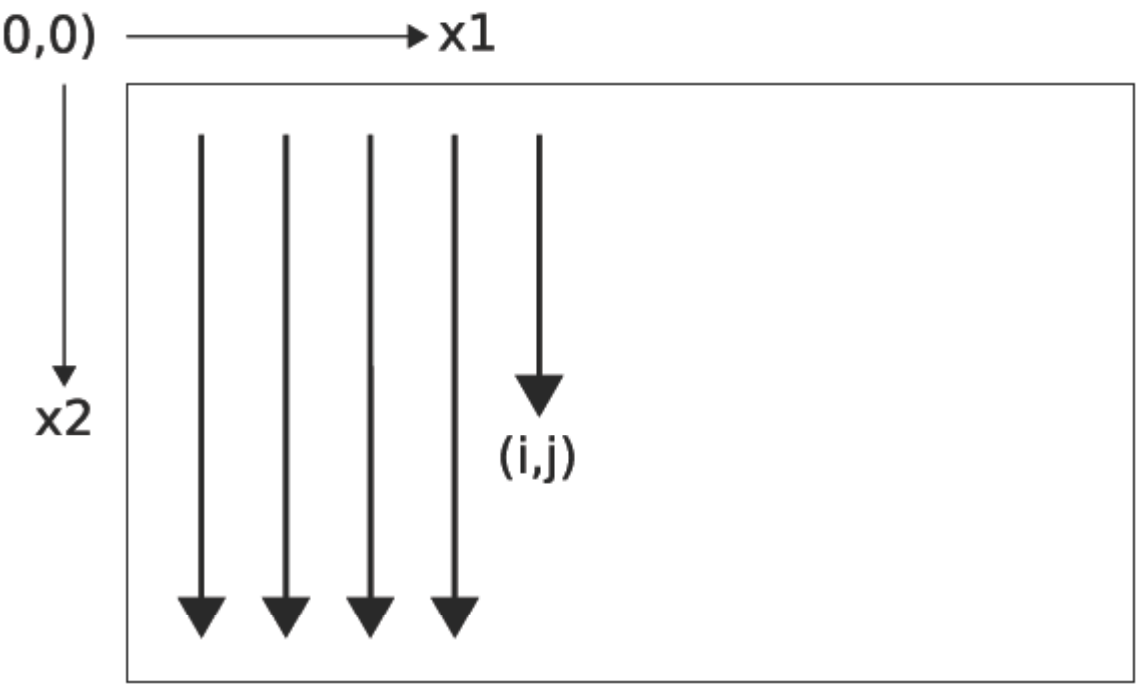
Discussion: BLSTM vs. Conventional HMM

- BLSTM are discriminative
- BLSTM allow correlated input features
- Internal states are continuous and multivariate, because they are defined by the vector of activations of the hidden units
- The output is a sequence of labels without duration information
- BLSTM is in principle able to access context from the entire input sequence

- Standard RNN requires one-dimensional input
- Idea (like DAG-RNN):
  - Each neuron receives external input and its own activation from one step back along all dimensions
  - Can be applied to any dimensional sequences (img 2D, video 3D)

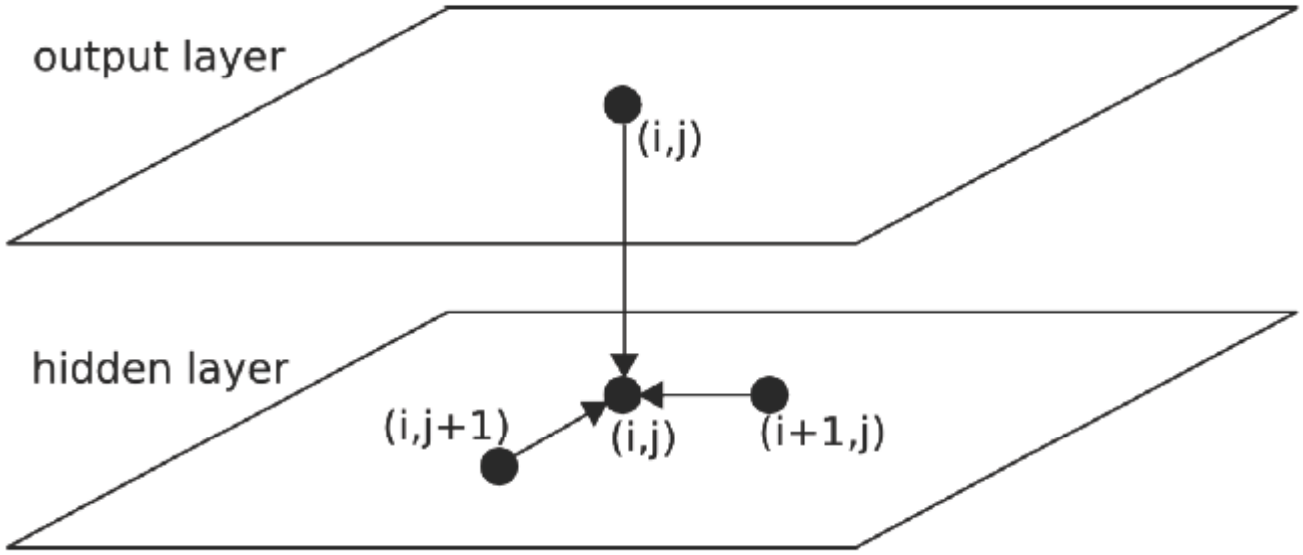


- Ensure that each previous' step output is already calculated
- Example:

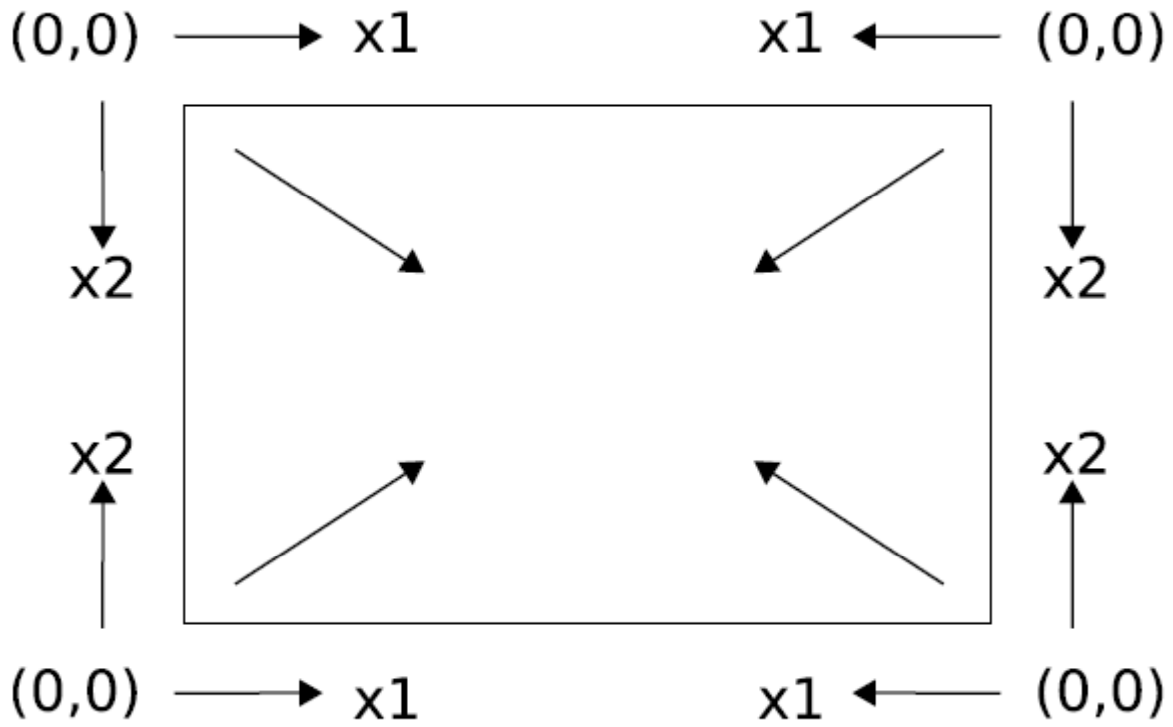


- Note: Boundaries have to be omitted

- N-dimensional backpropagation through time



- Idea: If d is the number of dimensions, use 2^d hidden layers
- Example (2 dimensions):

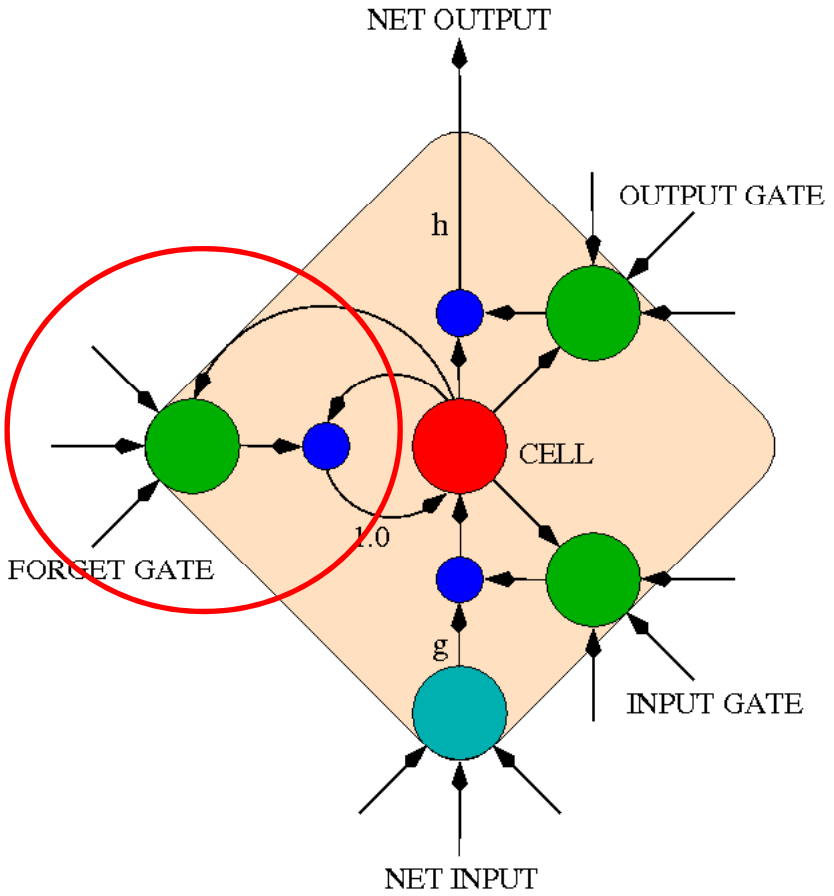


Problem

- Does the complexity explode?
- 2^d seems to be quite large
- However
  - Number of weights has more influence
  - Several calculations can be shared
- Furthermore
  - Reduce the size of the hidden layers with increasing dimensionality
  - It has been found that for speech recognition the number of weights was reduced to half and MDRNN gave still better results
- Main scaling concern is the size of the data, i.e., the length of the sequence

Combining idea with BLSTM

- Introduce 2^d self-connections, i.e., 2^d forget gates, each connected along one dimension
- However, only one input gate (connected to all dimensions)
- Also, only one output gate, since only the cell state is considered



Overall system

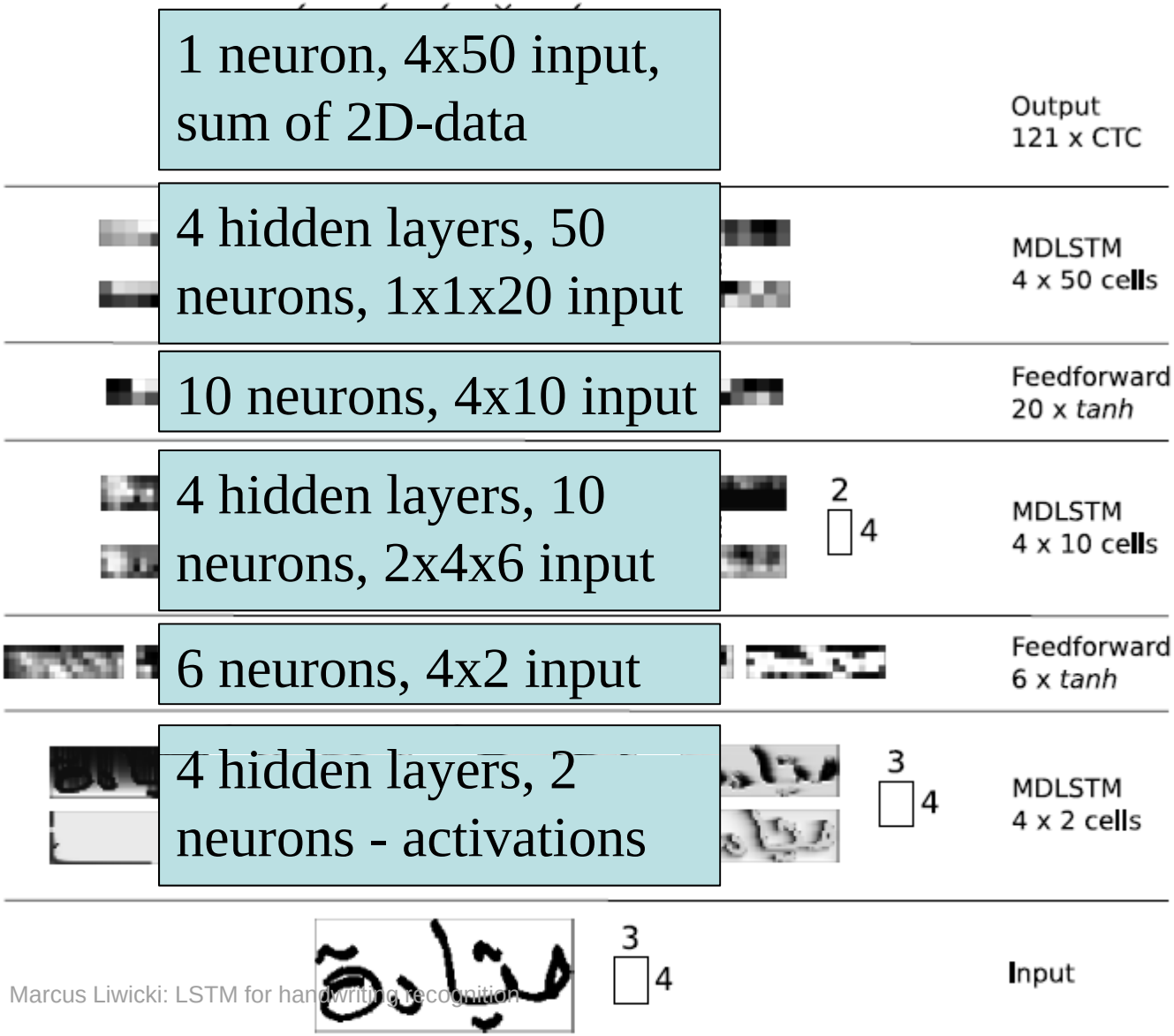
MDLSTM layers and feed-forward layers

4x3=>12-dim vector

Small at bottom large at top

159,369 weights but most at top

1D at top by summing up



Illustration

ICDAR 2007 Arabic handwriting recognition contest

System	SET f			SET s		
	top 1	top 5	top 10	top 1	top 5	top 10
CACI-3	14.28	29.88	37.91	10.68	21.74	30.20
CACI-2	15.79	21.34	22.33	14.24	19.39	20.53
CEDAR	59.01	78.76	83.70	41.32	61.98	69.87
MITRE	61.70	81.61	85.69	49.91	70.50	76.48
UOB-ENST-1	79.10	87.69	90.21	64.97	78.39	82.20
PARIS V	80.18	91.09	92.98	64.38	78.12	82.13
ICRA	81.47	90.07	92.15	72.22	82.84	86.27
UOB-ENST-2	81.65	90.81	92.35	69.61	83.79	85.89
UOB-ENST-4	81.81	88.71	90.40	70.57	79.85	83.34
UOB-ENST-3	81.93	91.20	92.76	69.93	84.11	87.03
SIEMENS-1	82.77	92.37	93.92	68.09	81.70	85.19
MIE	83.34	91.67	93.48	68.40	80.93	83.73
SIEMENS-2	87.22	94.05	95.42	73.94	85.44	88.18
Ours	**91.43**	**96.12**	**96.75**	**78.83**	**88.00**	**91.05**

Results of the ICDAR 2009 Arabic HWR Contest

System	Word Accuracy	Time/Image
CTC	81.06%	371.61 *ms*
Arab-Reader HMM	76.66%	2583.64 *ms*
Multi-Stream HMM	74.51%	143,269.81 *ms*

表 4 Summarized results from the offline Arabic handwriting recognition competition

Results of the ICDAR 2009 French HWR Contest

System	Word Accuracy
CTC	93.17%
HMM+MLP Combination	83.17%
Non-Symmetric HMM	76.34%

Summarized results from the offline (French) handwriting recognition competition

And the Very Best

- It is open source and for free
- http://github.com/mrhichem/RNNLIB
- Examples are online (Arabic recognition)

Thank You

DFKI, Kaiserslautern

Marcus.Liwicki@dfki.de

Further Reading:

Marcus Liwicki and Horst Bunke: Handwriting Recognition for Whiteboard Notes – Online, Offline, and Combination (World Scientific, 2008)

Alex Graves, Marcus Liwicki, et al. A novel connectionist system for unconstrained handwriting recognition", IEEE TPAMI, 31, 5, pp. 855-868 (2009).

Alex Graves' PhD thesis: Supervised Sequence Labelling with Recurrent Neural Networks (2008)

A. Graves and J. Schmidhuber. "Offline handwriting recognition with multidimensional recurrent neural networks". Advances in NIP Systems 21, pages 545-552 (2009).