technische universität
dortmund

Deep Learning for Word Spotting

— ICFHR 2018 Tutorial, Niagara Falls, USA —

Gernot A. Fink

August 5, 2018

- Introduction
- Fundamentals: Bag-of-Features
- Learning Document Image Representations
- Learning Word Spotting Models
- Deep Learning Fundamentals
- Deep Learning for Word Spotting
- Summary

with contributions by Sebastian Sudholt, Leonard Rothacker, René Grzeszick

Introduction: Automatic Reading Systems

State of Automatic Reading:

- One of the earliest application fields studied in computer science
- So-called OCR achieves high-quality results for machine-printed text in well-defined settings.
- Online handwriting recognition again gaining popularity
- Offline handwriting recognition: Remarkable results, but still an open research problem

General Methodology:

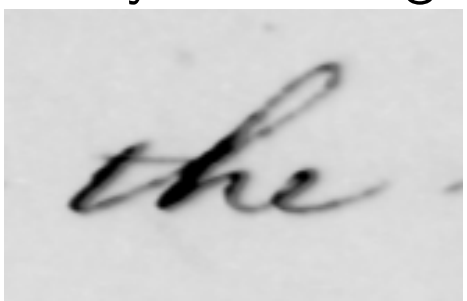
Statistical sequence models (e.g. HMMs, BLSTMs) that are trained from *extensive* amounts of example data

Introduction: Why Word Spotting?

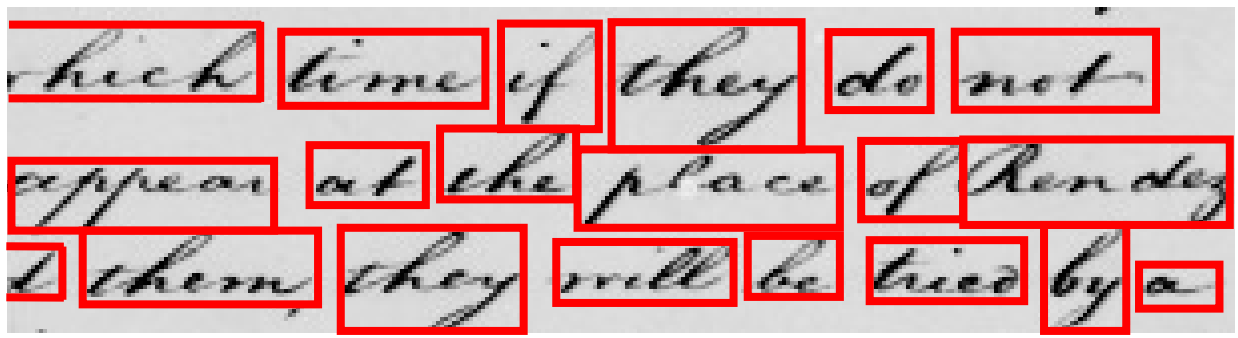
What if automatic transcription of handwriting is no longer feasible?

Alternative: Retrieval of individual words rather than transcription ("query-by-example")

Query word image



Document image

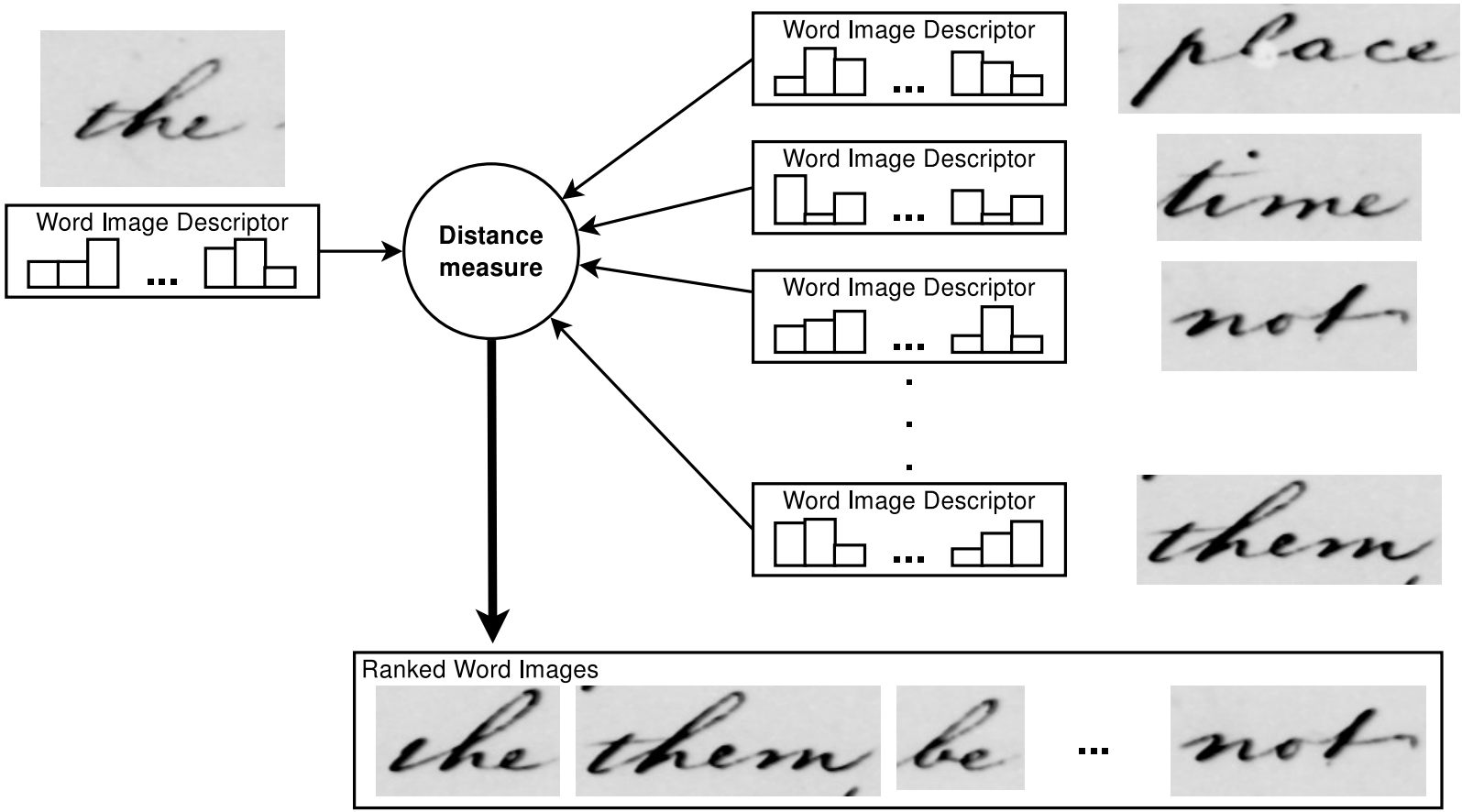


Images from *The George Washington Papers at the Library of Congress, 1741-1799*

Introduction: Basic Methodology

Query-by-example word spotting

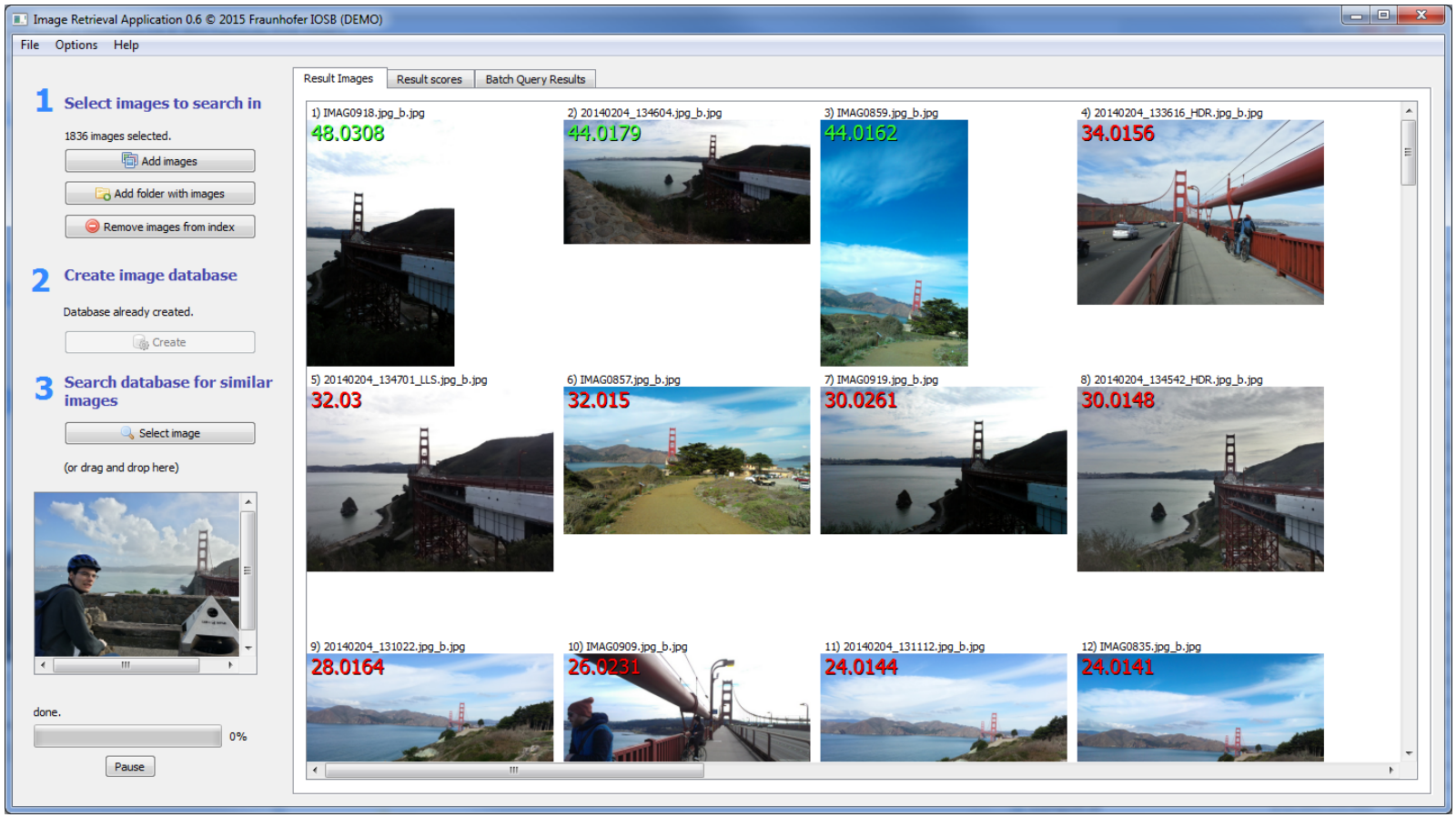
Does that seem familiar?



Based on [Rath & Manmatha, IJRAR'07]

Introduction: Basic Methodology II

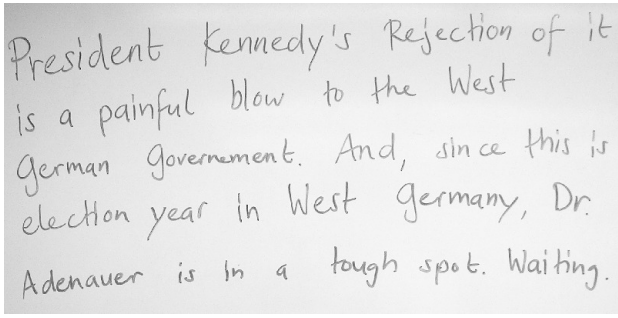
QbE word spotting ≈ special case of **content-based image retrieval**



(Source: Fraunhofer IOSB CBIR Demo)

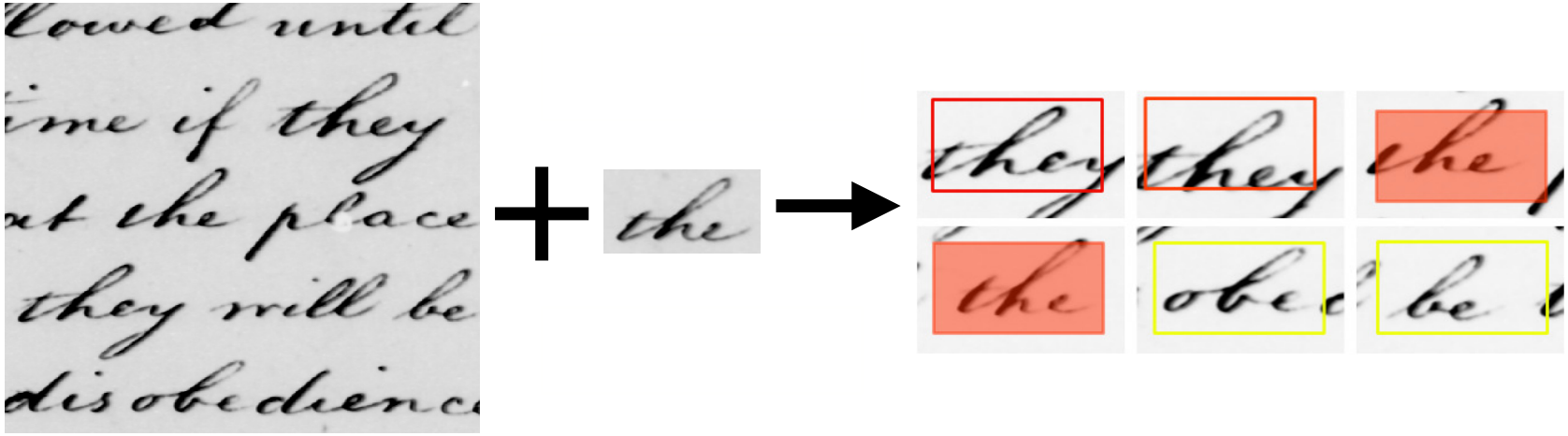
Tasks in Handwriting Recognition

Document Transcription (= "classical" recognition)



President Kennedy is Rejection of it is a painful blow to the West German Government And since this is selection year in West Germany of Adenauer is in a tough spot waiting

Document Retrieval (aka "Word Spotting")



Word Spotting: Fundamentals

Core Methodology:

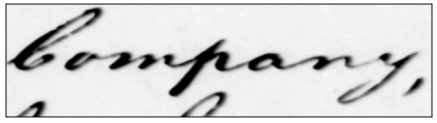
- Specialized image retrieval
- Important ingredient: Image matching procedure
- Frequently required: Pre-segmentation (words / lines)

Taxonomy:

- *Segmentation-based*
- *Segmentation-free*, i.e., segmentation problem covered during retrieval
- *Query-by-Example*, i.e., word image directly used as query
- *Query-by-String*, i.e., query model derived from textual query ("string")

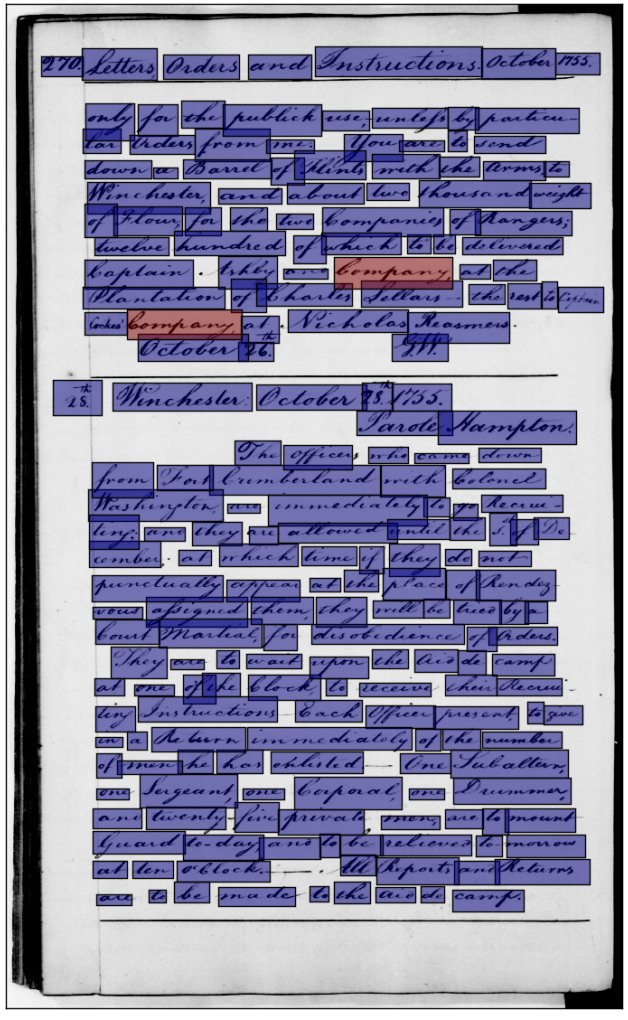
Word Spotting Tasks

Query by Example



Query by String

Company



Images from *The George Washington Papers at the Library of Congress, 1741-1799*

Word-Spotting: Classical Milestones

Manmatha *et al.* 1996: First influential work

(Binarization, Alignment, XOR distance)

Rath & Manmatha 2003: DTW matching

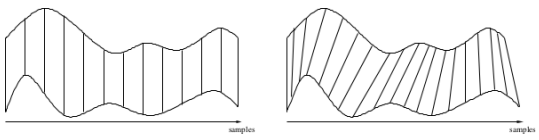
(Normalization, profile features)

Rusiñol *et al.* 2011: First influential work using BoF, first with Spatial Pyramid

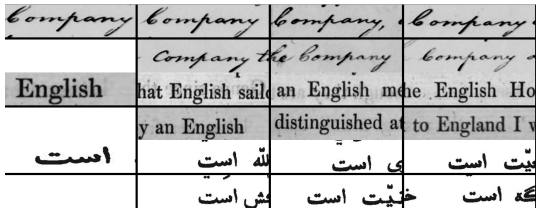
(SIFT, BoF, Spatial Pyramid, LSI, segmentation-free decoding)

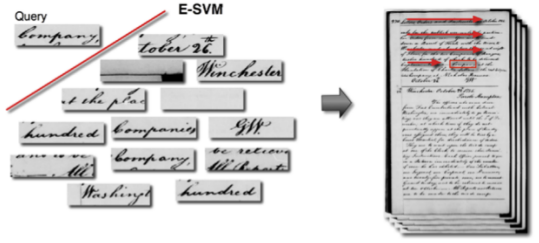
Almazan *et al.* 2014: HOG features

("Exemplar SVM", query expansion)



(a) naive alignment after resampling, (b) alignment with DTW.





Overview

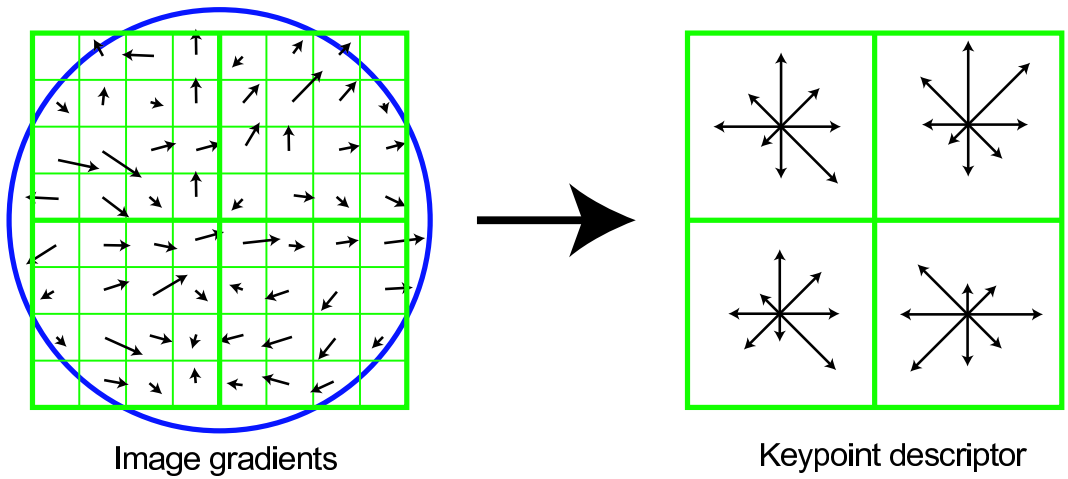
- Introduction
- Fundamentals: Bag-of-Features
- Learning Document Image Representations
- Learning Word Spotting Models
- Deep Learning Fundamentals
- Deep Learning for Word Spotting
- Summary

Local Image Descriptors

Fundamentals:

- Local gradient statistics
  (i.e. implicit description of object contours)
- Grouping of multiple such local statistics in a certain neighborhood and *normalization*
  (coarse description of structural properties)

Example: SIFT descriptor [Lowe, 2004]



(Source: [Lowe, IJCV 2004])

Note: Ex. several similar descriptors, e.g., HOG, SURF

Lowe, D: *Distinctive Image Features from Scale-Invariant Keypoints*, Int. J. of Computer Vision, 60(2), pp. 91–110, 2004.

Statistical Image Modeling

Representation: Bag-of-Features Models (BoF)

- Originally proposed as Bag-of-Words models for representing texts

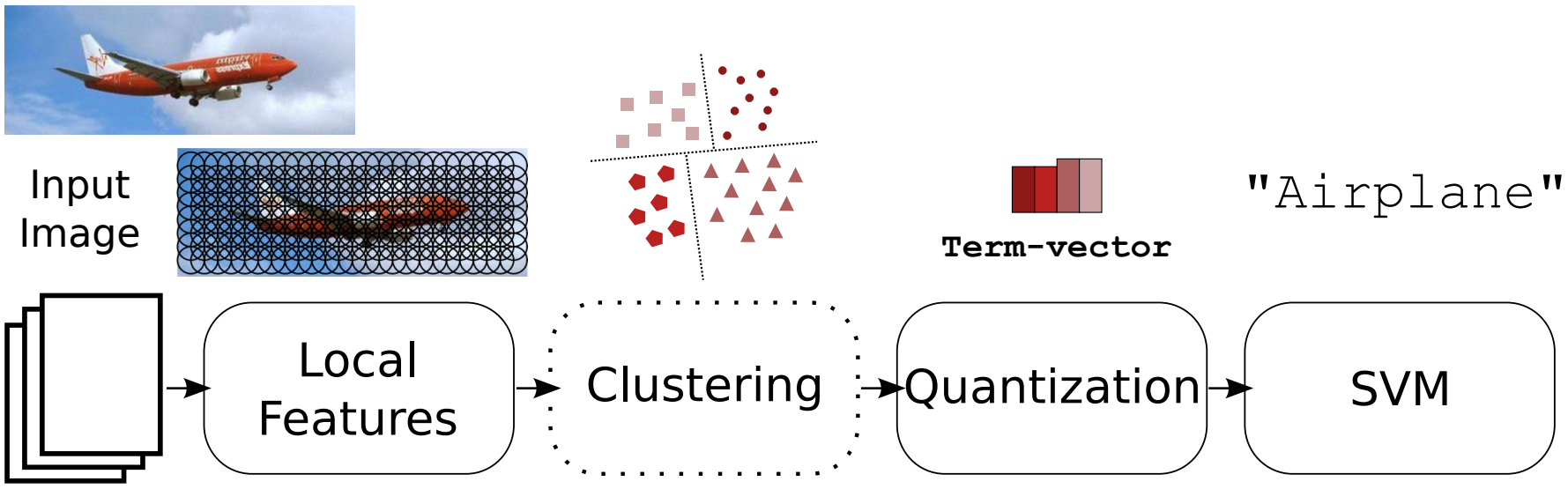
BoF-Approach: (cf. e.g. [O'Hara & Draper 2011])

0. Extract local image features (usually in a grid)
1. Compute *visual vocabulary* by quantizing local image features
2. For given image, compute histogram of quantized features (i.e. orderless "bag" of features)
3. Any classification / matching technique can be applied to BoF representations

S. O'Hara & B. A. Draper: Introduction to the Bag of Features Paradigm for Image Classification and Retrieval, Computing Research Repository, arXiv:1101.3354v1, 2011.

Statistical Image Modeling II

BoF-Approach: Processing Pipeline

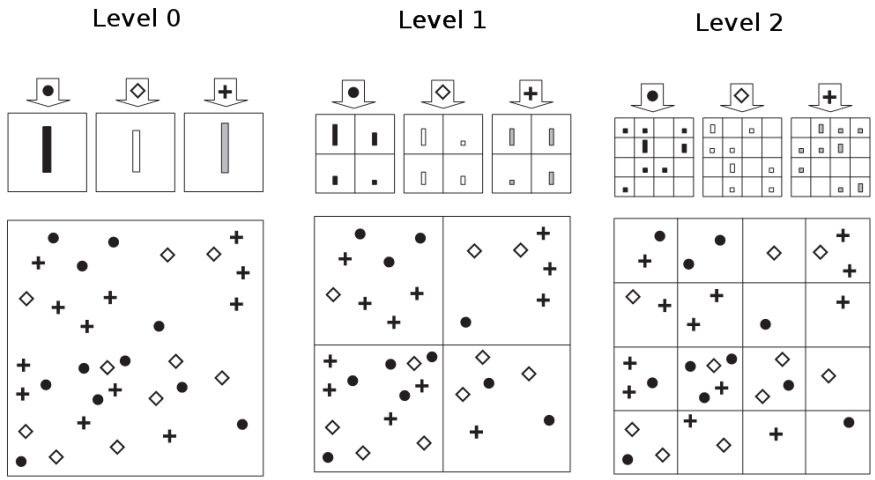


Main drawback: Spatial information is lost!

Statistical Image Modeling III

Goal: Compensate loss of spatial information

Method: Spatial-pyramid models



Lazebnik, S., Schmid, C., Ponce, J.: *Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories*, IEEE Conference on Computer Vision and Pattern Recognition, 2006.

Disadvantage: Considerably increased model complexity

Overview

Bag-of-Features Models for Word Spotting

Basic Methodology:

Image Features:

Gradient-based descriptors, e.g., SIFT, HOG

Feature Extraction:

Dense grid, i.e., no keypoint detection involved

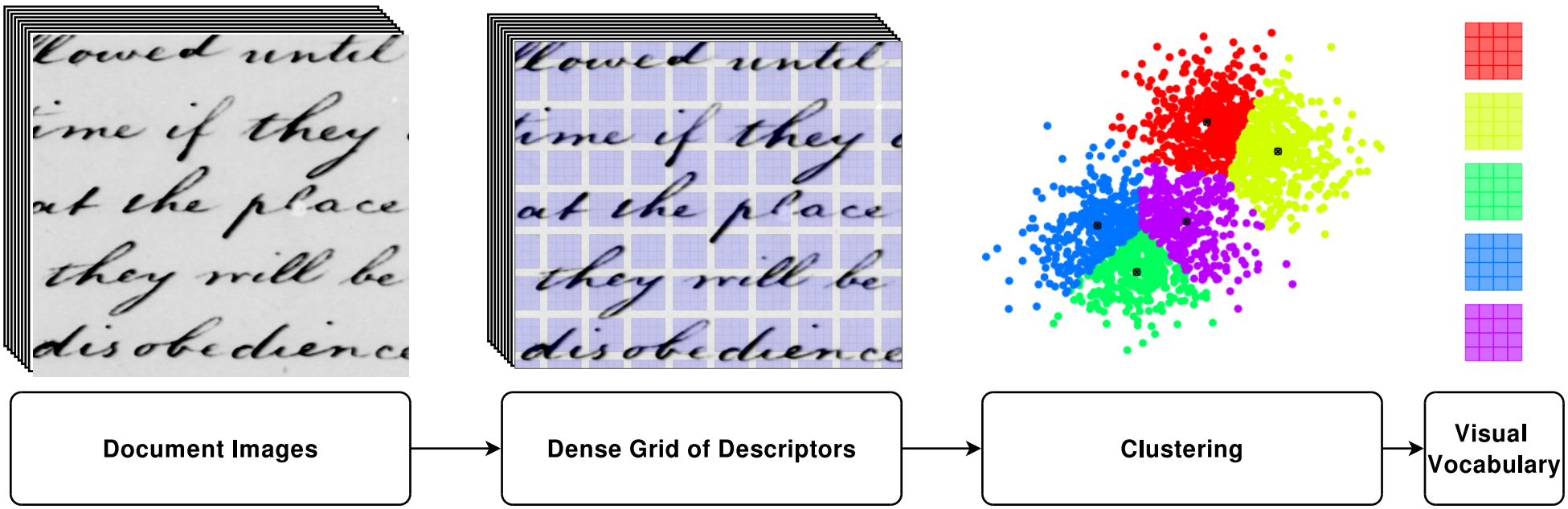
Visual Vocabulary:

- Quantization of descriptors (as "usual")
- Special: Large vocabularies (i.e. 2K to 4K) in order to capture appearance of "individuals" precisely

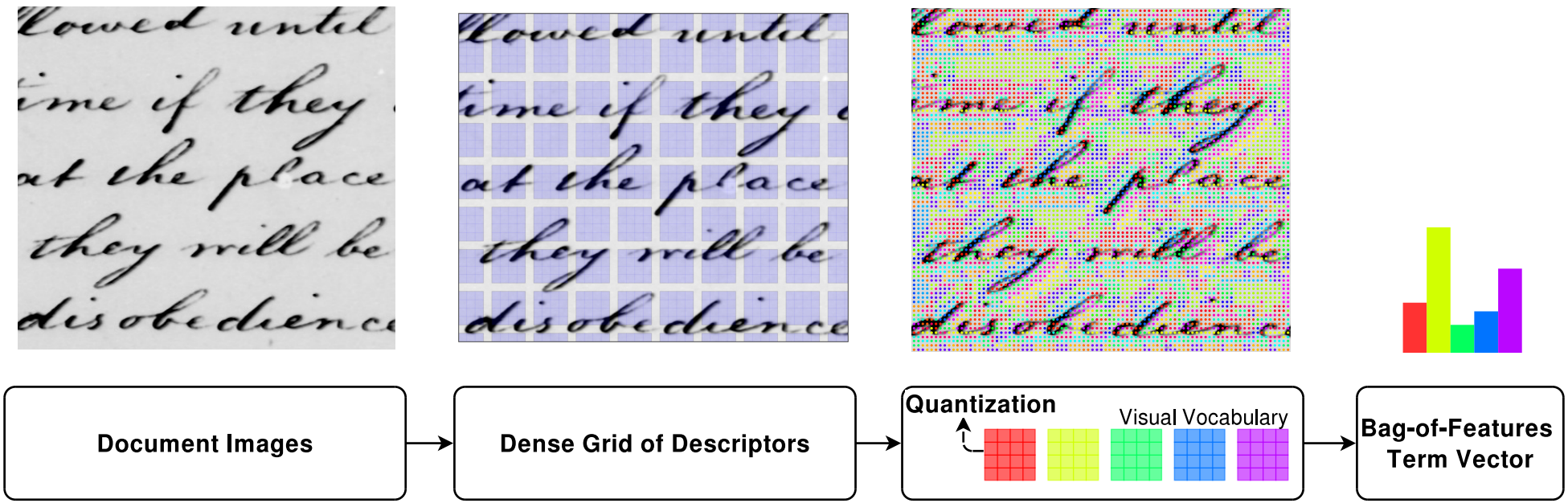
Query Model:

- Segmentation- or patch-based processing
- 1D spatial pyramid for improved spatial modeling

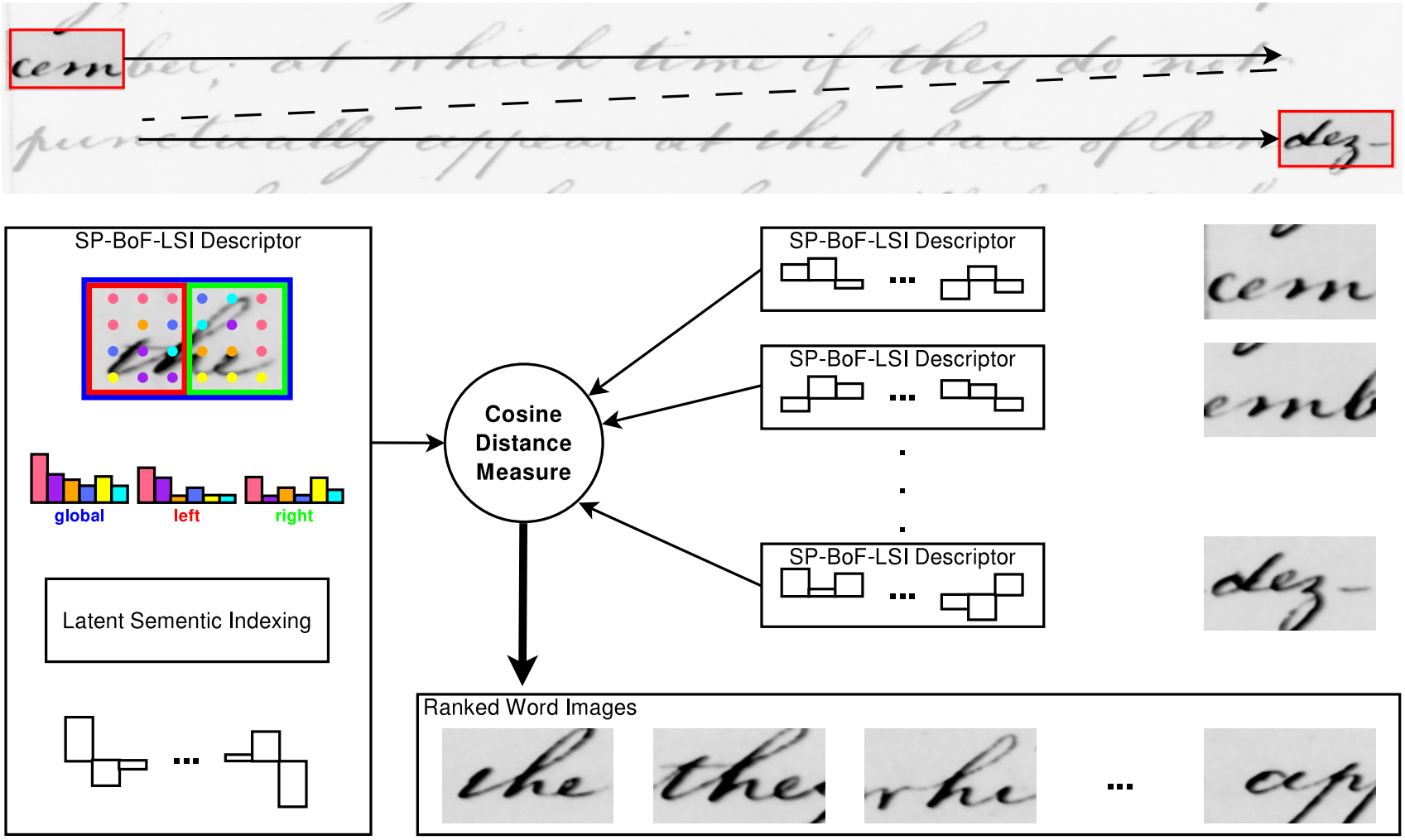
Bag-of-Features Models: Visual Vocabulary



Bag-of-Features Models: Term Vector

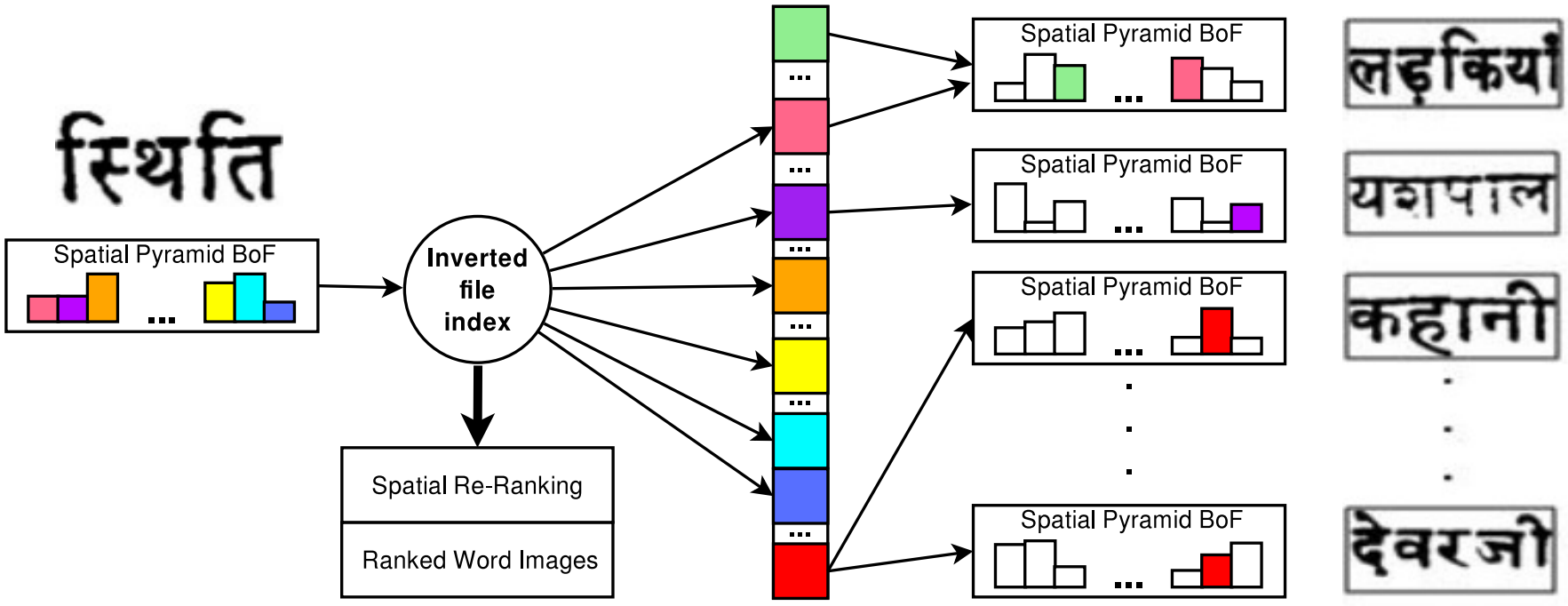


Segmentation-free Word Spotting with Bag-of-Features



Rusiñol, M., and Aldavert, D., and Toledo, R., and Lladós, J.: Browsing heterogeneous document collections by a segmentation-free word spotting method, Int. Conf. on Document Analysis and Recognition, Bejing, pp. 63-67, 2011.

Segmentation-based Word Spotting with Bag-of-Features



Shekhar, R., and Jawahar, C.: Word image retrieval using bag of visual words, Int. Workshop on Document Analysis Systems, pp. 297-301, 2012.

Overview

Learning BoF-Based Word Spotting Models

What we have so far:

- Expert-designed image descriptors
- Automatically learned document image features
  (visual vocabulary $\rightarrow$ histograms of quantized descriptors)
- Matching: Nearest neighbor

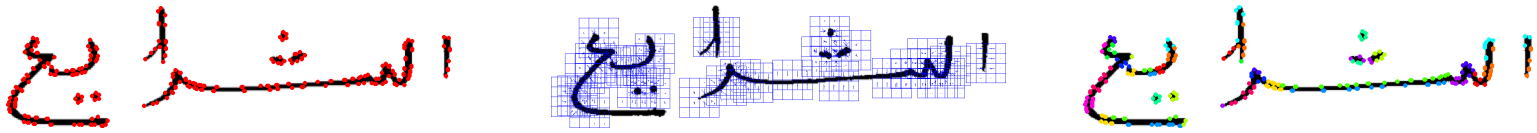
What we want in addition:

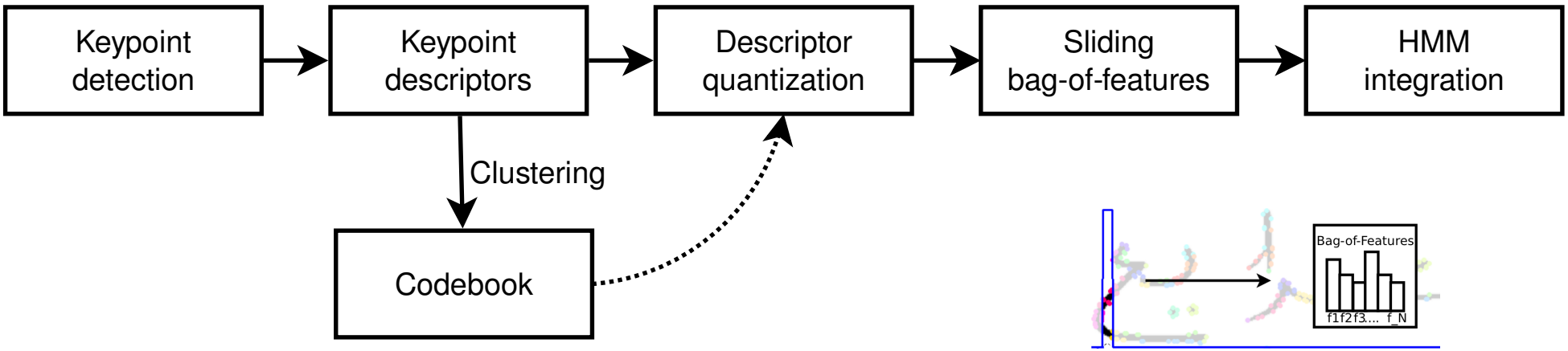
- More powerful image matching
- Categorization capabilities
  (i.e. links between image appearance and textual representation)

What we need: Learned classification models

Bag-of-Features HMMs

- *Extension* of HMMs towards learned feature representation
- *Extension* of BoF models towards fine-grained script representation

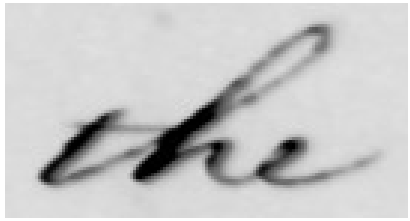




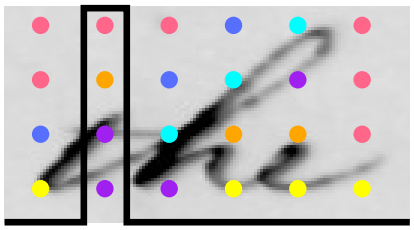
$$b_j(\boldsymbol{f}) = \sum_{k=1}^{|V|} c_{jk} f_k \quad \text{with} \quad \begin{array}{l} \boldsymbol{f} : \text{term vector} \\ V : \text{vis. voc.} \end{array}$$

Rothacker, L., Vajda, S., Fink, G. A.: *Bag-of-Features Representations for Offline Handwriting Recognition Applied to Arabic Script*, In Proc. ICFHR, Bari, 2012.

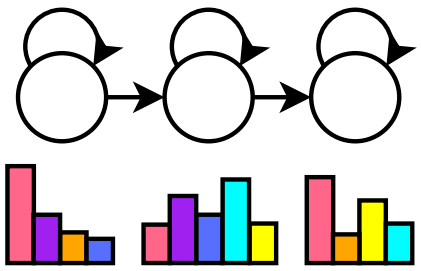
Bag-of-Features HMMs for QbE Word Spotting



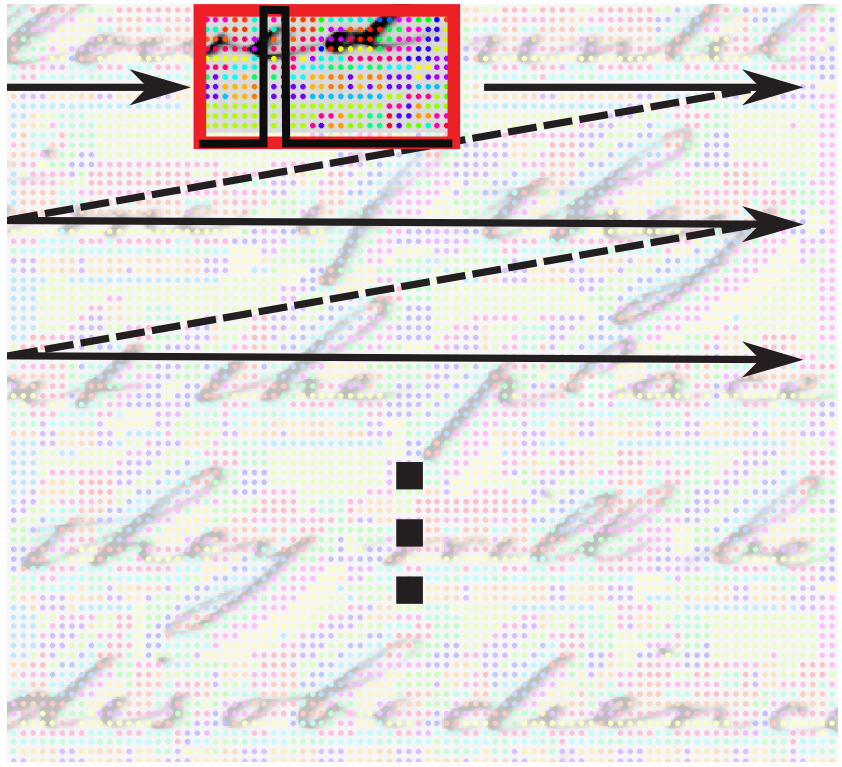
Query Word



Bag-of-Features Sequence
(sliding window)



Bag-of-Features
Hidden Markov Model

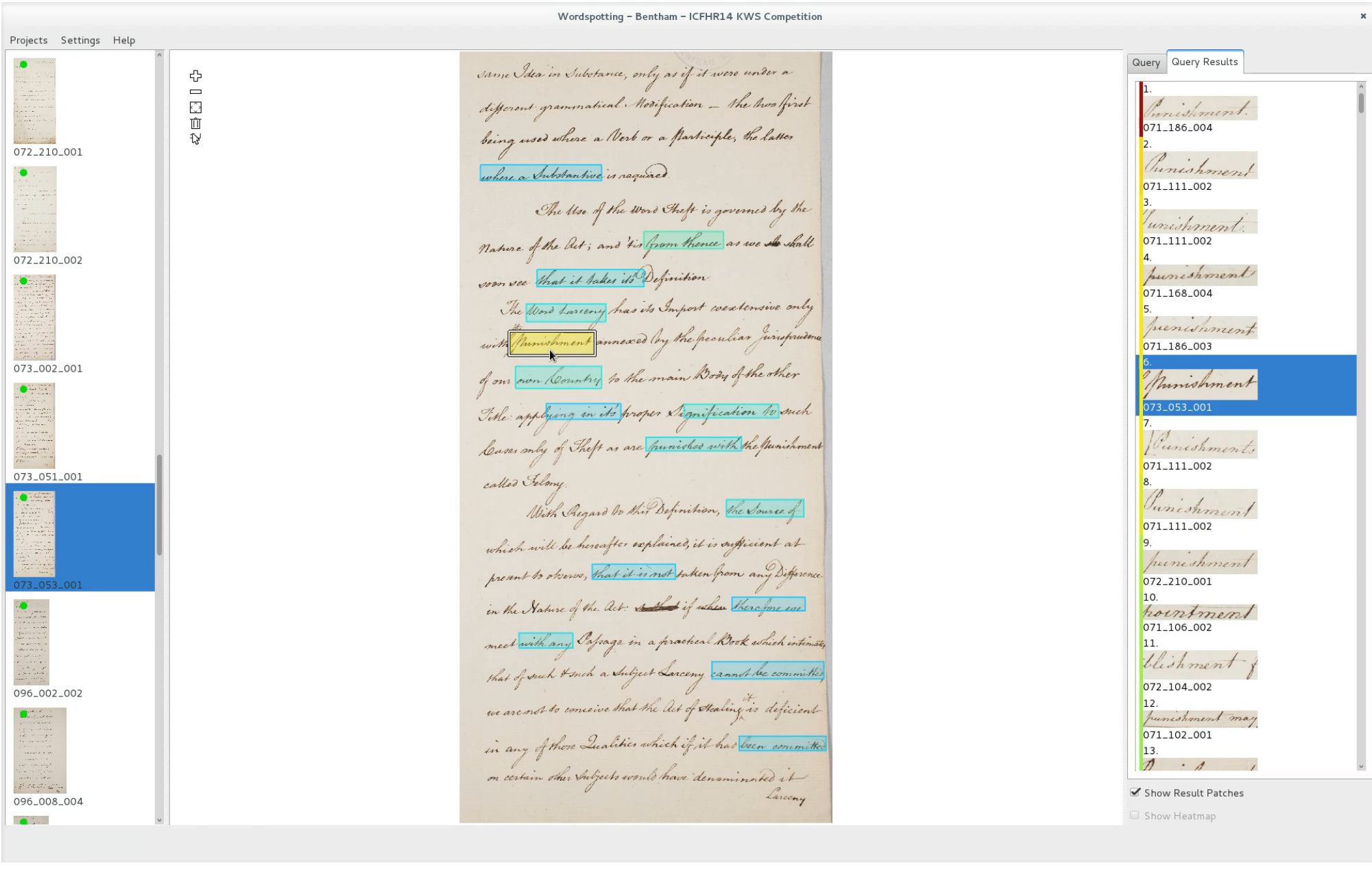


- Approach can be sped-up by intelligent patch pre-filtering!

Rothacker, L., Rusinol, M., Fink, G. A.: *Bag-of-Features HMMs for Segmentation-Free Word Spotting in Handwritten Documents*, In Proc. Int. Conf. on Document Analysis and Recognition, Washington DC, USA, 2013.

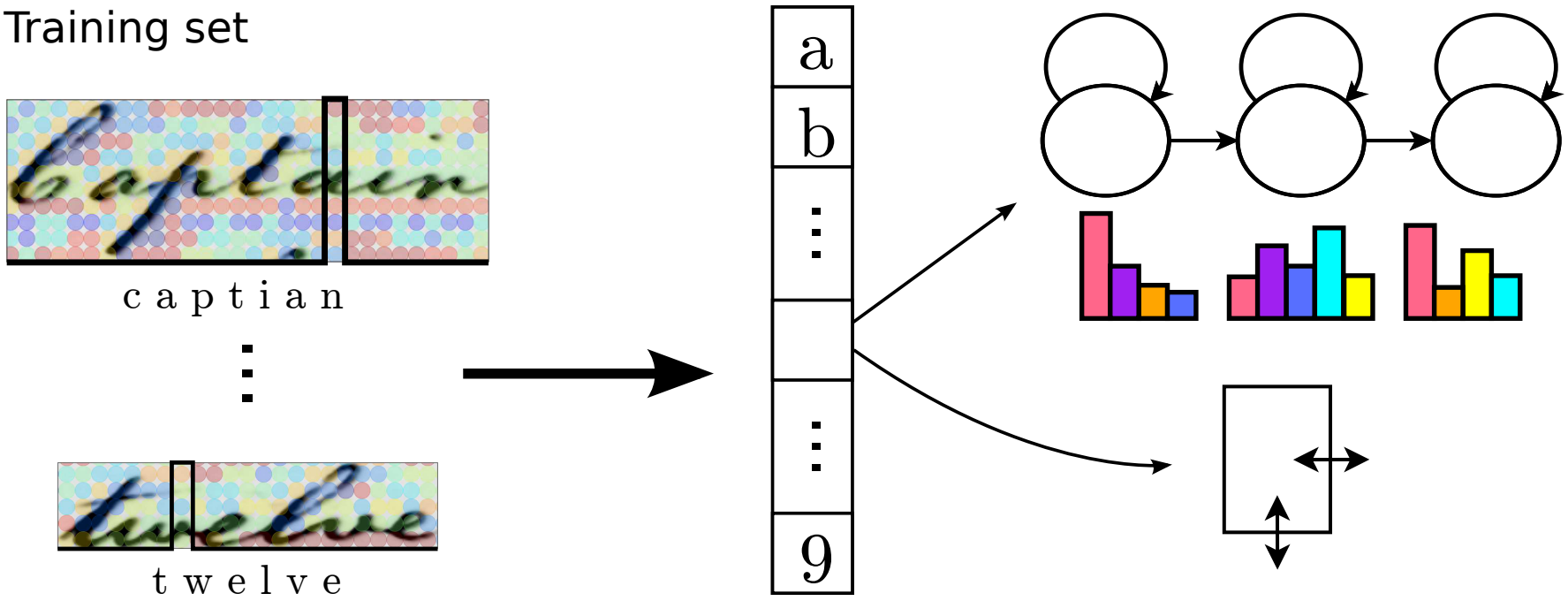
Rothacker, L., Rusinol, M., Llados, J., Fink, G. A.: *A Two-Stage Approach to Segmentation-Free Query-by-Example Word Spotting*, manuscript cultures, 1(7), pages 47-57, 2014.

QbE Word Spotting: Example ("Bentham")



(Video created by Matthias Kasperidus)

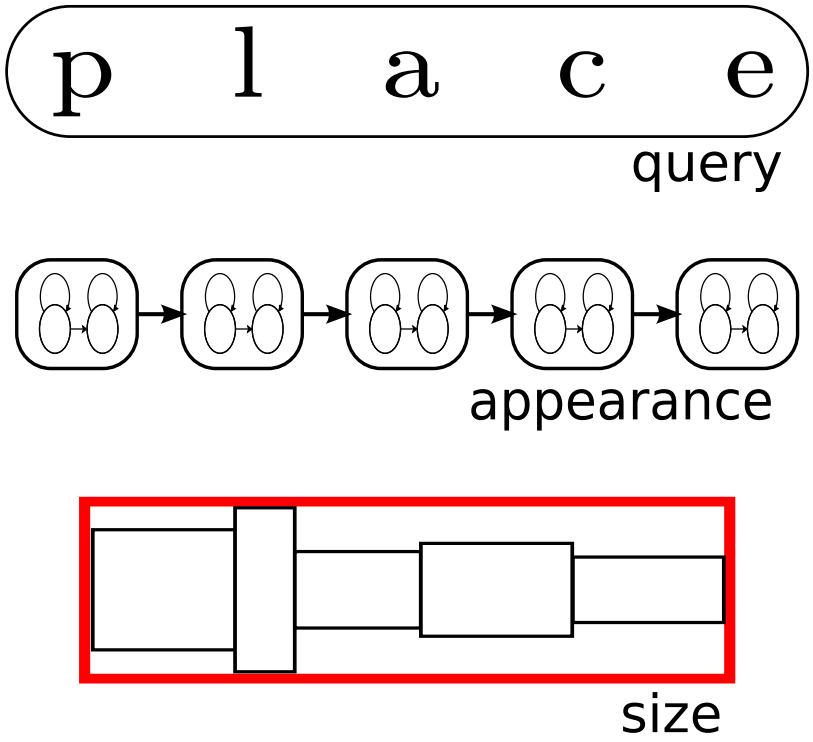
Bag-of-Features HMMs for QbS Word Spotting

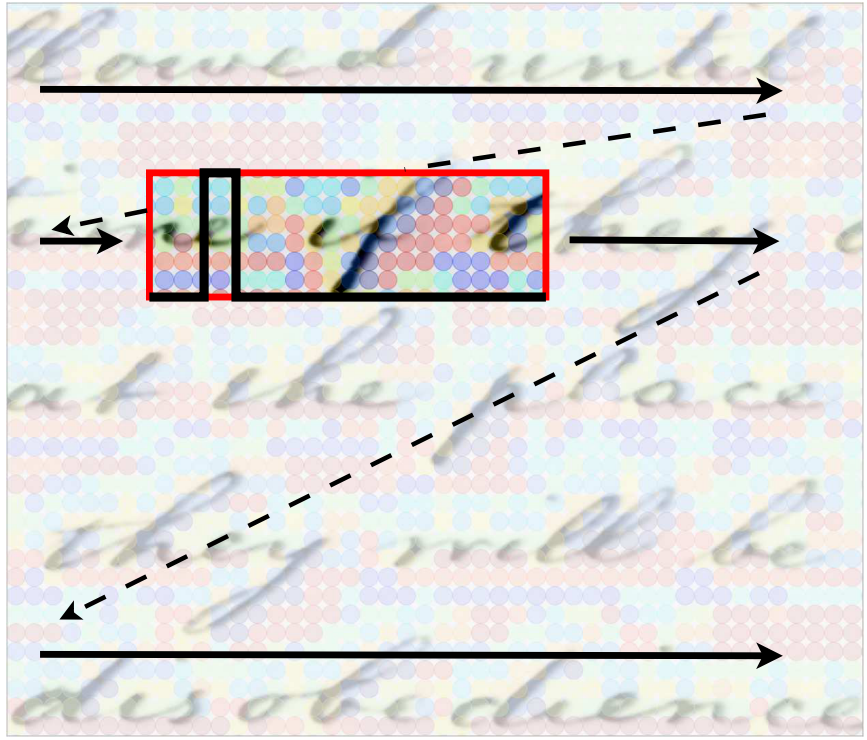


- Appearance model: Bag-of-Features HMMs (per character)
- Spatial size model: Width & height estimates

Rothacker, L., Fink, G. A.: *Segmentation-Free Query-by-String Word Spotting with Bag-of-Features HMMs*, Int. Conf. on Document Analysis and Recognition, Nancy, France, 2015.

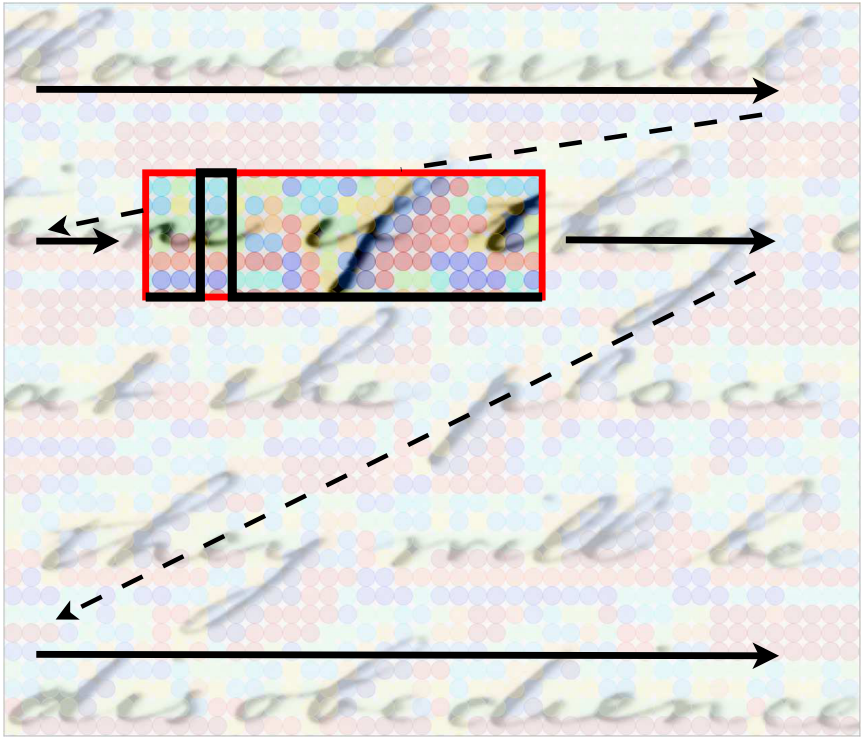
Bag-of-Features HMMs for QbS Word Spotting II

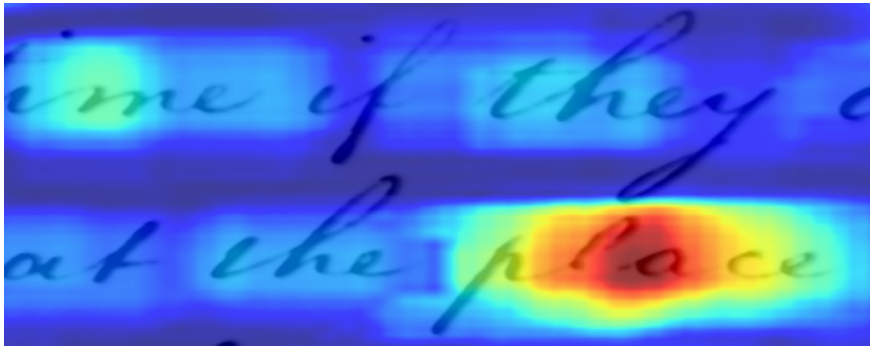


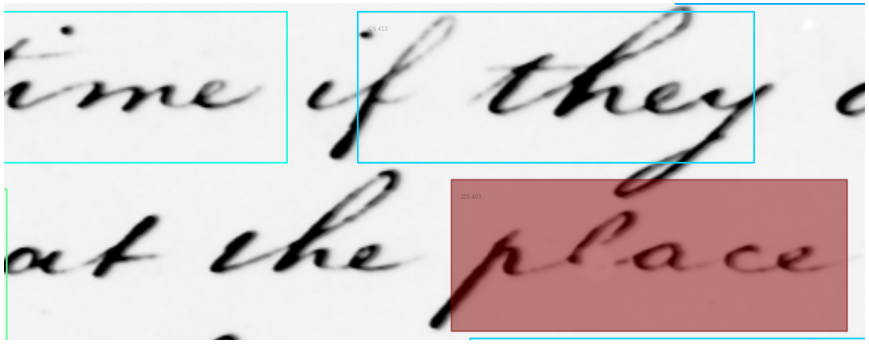


Rothacker, L., Fink, G. A.: *Segmentation-Free Query-by-String Word Spotting with Bag-of-Features HMMs*, Int. Conf. on Document Analysis and Recognition, Nancy, France, 2015.

Bag-of-Features HMMs for QbS Word Spotting II





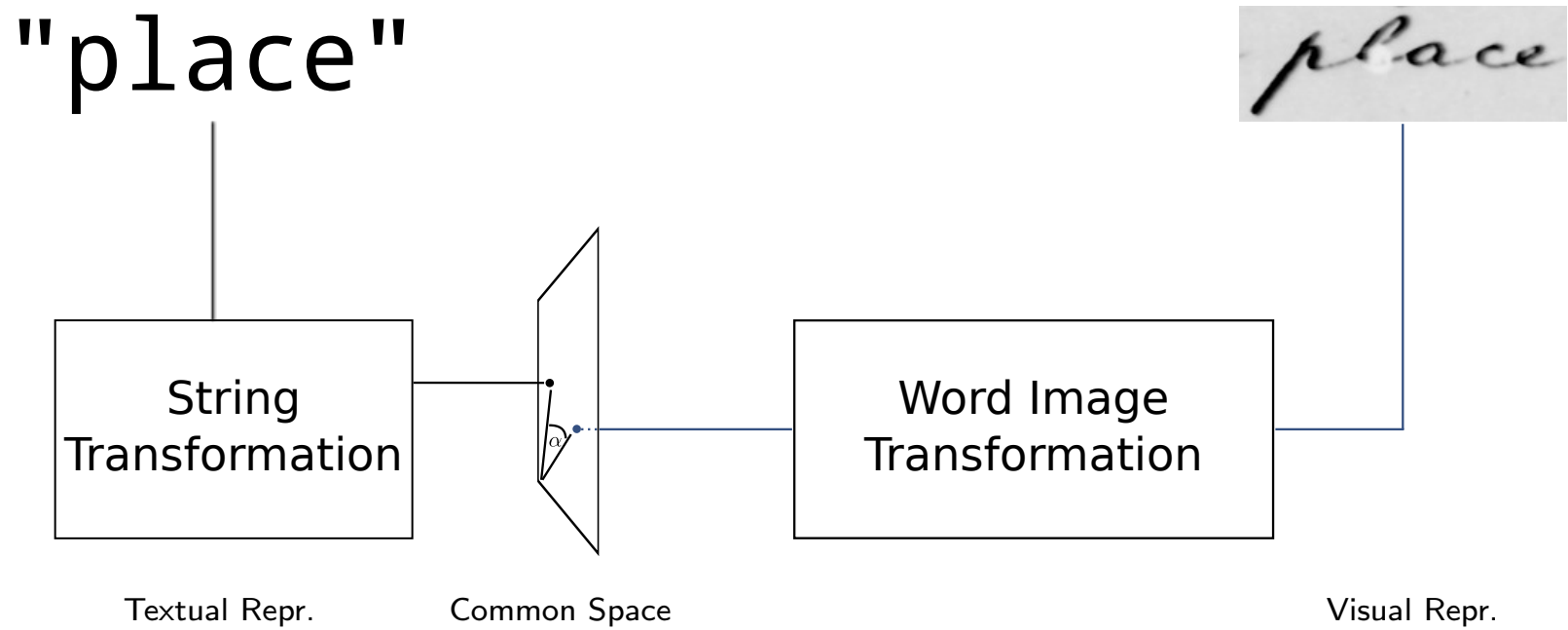


Rothacker, L., Fink, G. A.: *Segmentation-Free Query-by-String Word Spotting with Bag-of-Features HMMs*, Int. Conf. on Document Analysis and Recognition, Nancy, France, 2015.

Subspace Representations for Word Spotting

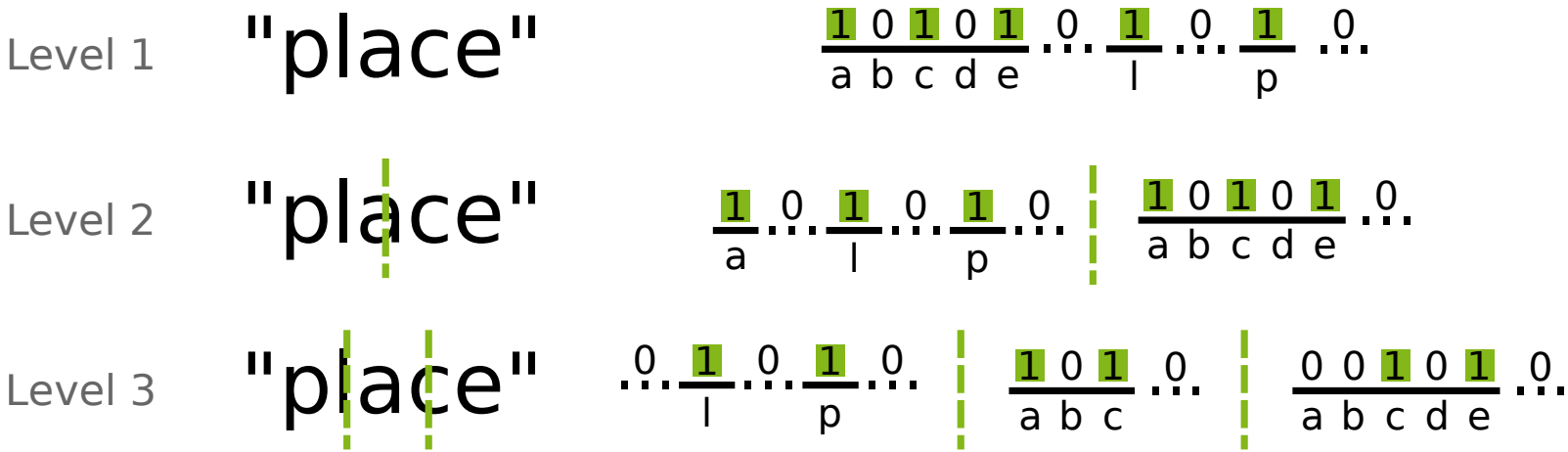
Idea: Project both textual and visual representation into a *common* space

Benefits: QbE and QbS are now a simple nearest neighbor search



J. Almazán, A. Gordo, A. Fornés and E. Valveny: Word Spotting and Recognition with Embedded Attributes, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 36, no. 12, pp. 2552-2566, 2014.

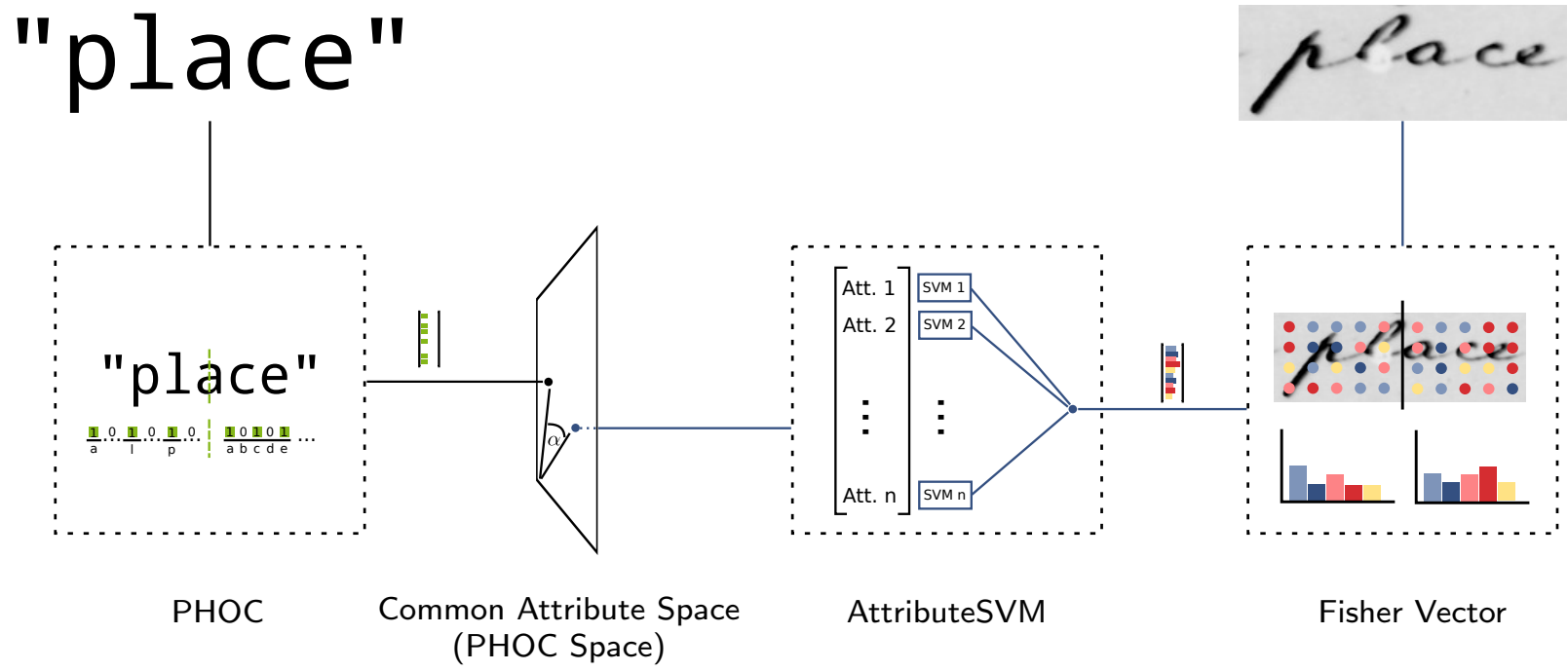
Pyramidal Histogram of Characters (PHOC)



- Concatenate histograms for all levels to form PHOC
- Levels used by Almazán *et al.*: 2,3,4 and 5
- 26 Characters + 10 Digits
- PHOC $\in \{0,1\}^{604}$

J. Almazán, A. Gordo, A. Fornés and E. Valveny: Word Spotting and Recognition with Embedded Attributes, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 36, no. 12, pp. 2552-2566, 2014.

Learning the PHOC representation



- AttributeSVM: ensemble of SVMs
- each SVM predicts one attribute within the PHOC

J. Almazán, A. Gordo, A. Fornés and E. Valveny: Word Spotting and Recognition with Embedded Attributes, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 36, no. 12, pp. 2552-2566, 2014.

Pros and Cons of Methods so Far

Basic BoF-Based Models:

- ✓ (partly) Learned image features / representations
- ✓ No annotations required (unsupervised)
- ↯ Purely image-based (no categorial information)

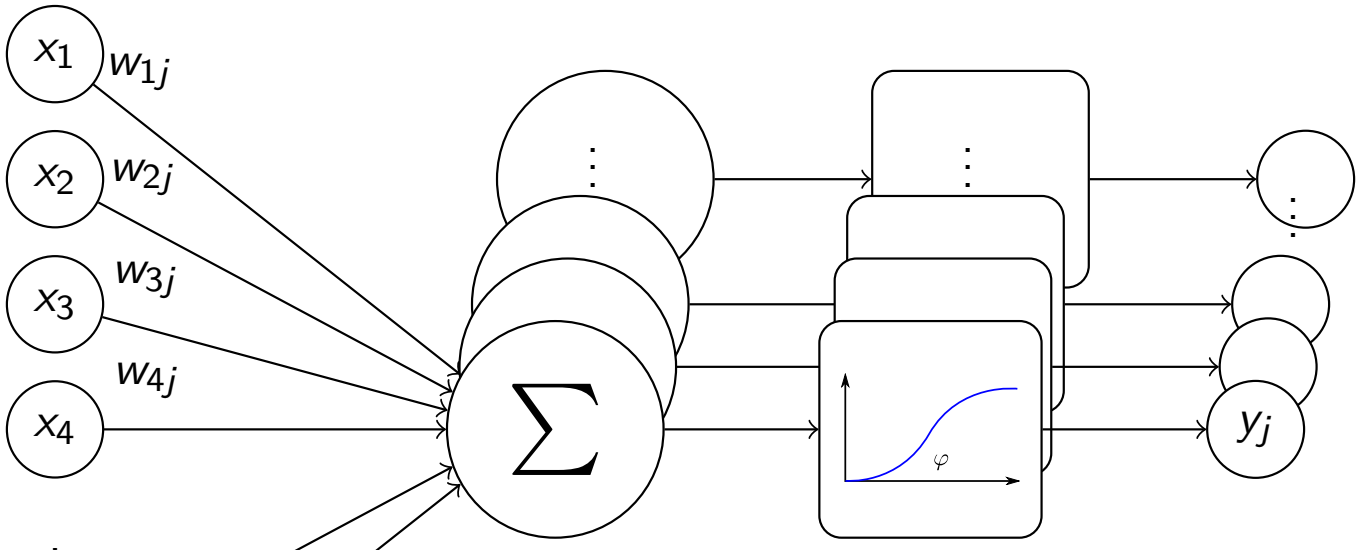
Advanced BoF-Based Models (BoF-HMMs, AttributeSVMs):

- ✓ Learned (categorial) models (supervised)
- ✓ Common representation of image appearance and categorial information (embedded attributes)
- ↯ Features and model learned separately

Desired: Integrated framework with overall / end-to-end optimization

Overview

The Perceptron



Single Output

$$y_j = \varphi\left(\sum_i w_{ij} x_i + b_j\right)$$

$$\mathbb{R}^n \rightarrow \mathbb{R}^m$$

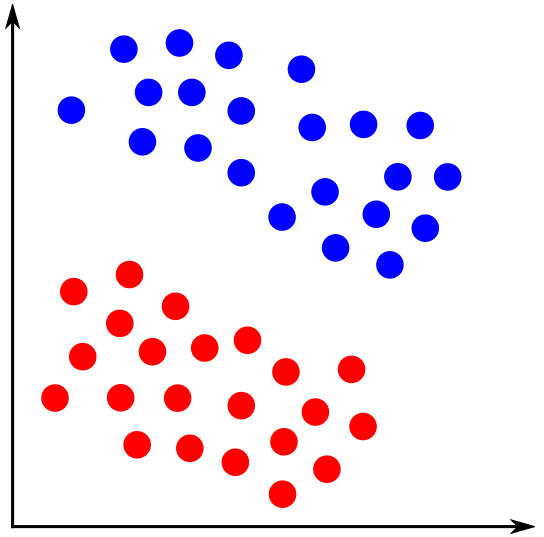
$$\mathbf{y} = \varphi\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right)$$

F. Rosenblatt: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Psychological Review, 65(6), 1958.

Capabilities of the Perceptron

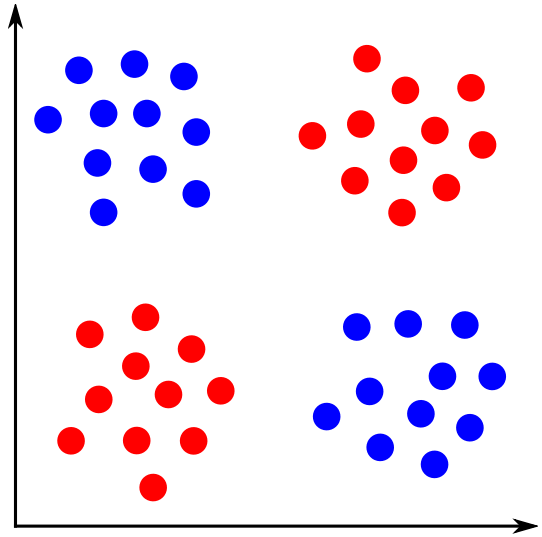
What a Perceptron can do:

Calssify two linearly separable classes



What a Perceptron can't do:

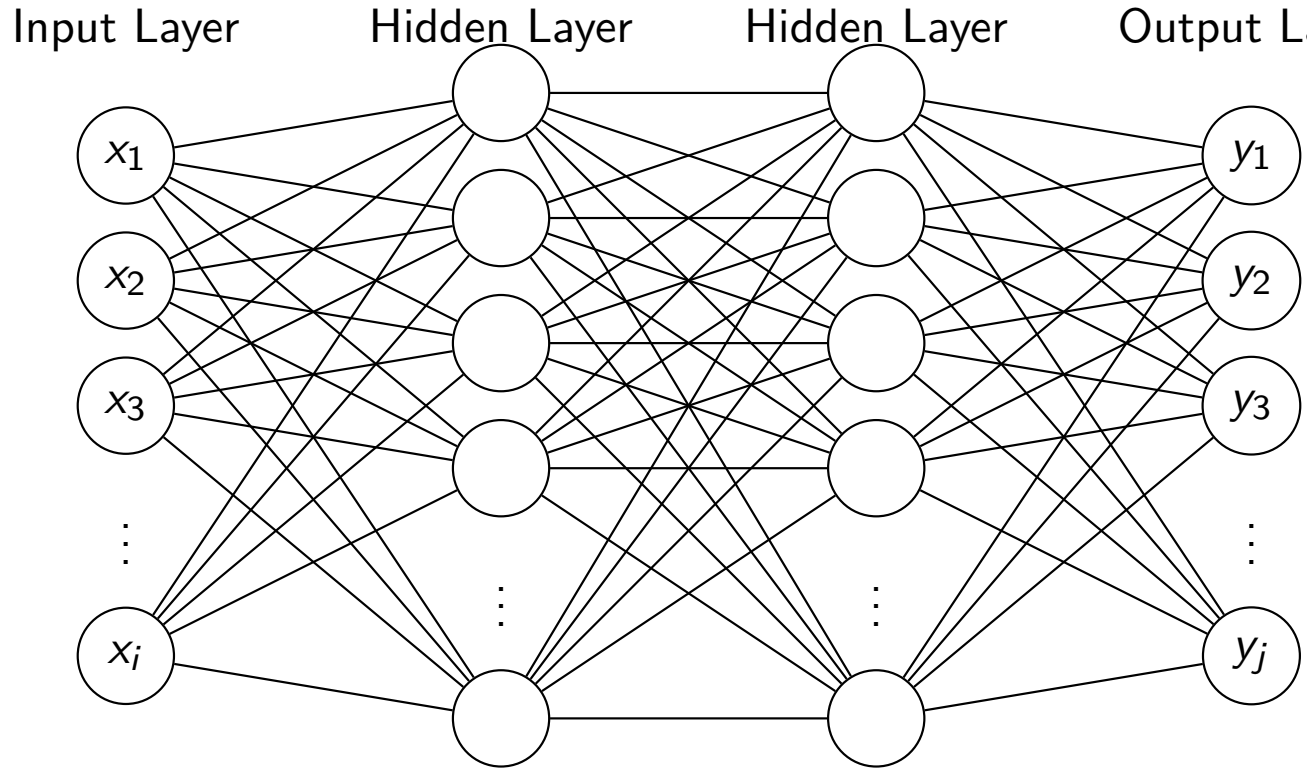
Classify two non-linearly separable classes (e.g. XOR-Problem)



Solution: Stack layers of Perceptrons

$\Rightarrow$ Multi Layer Perceptron

Multi Layer Perceptron (MLP)



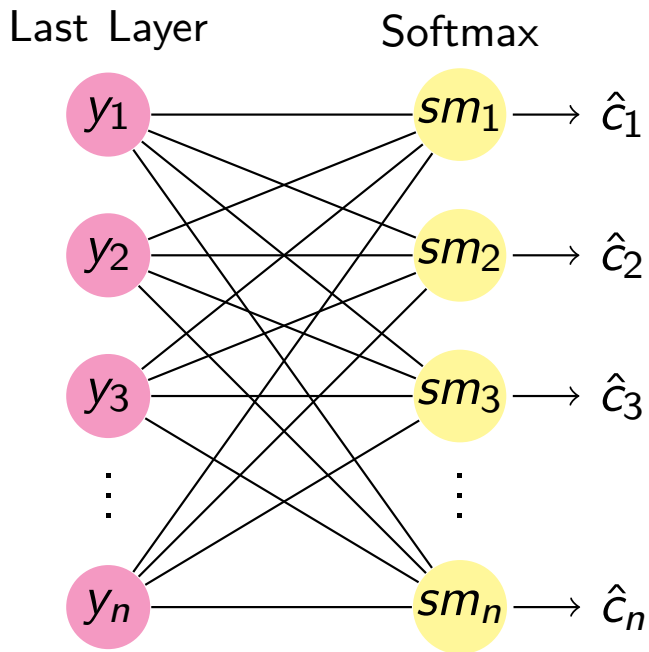
$$\mathbf{y} = \boldsymbol{f}^L\left(\boldsymbol{f}^{L-1}(\ldots \boldsymbol{f}^2\left(\boldsymbol{f}^1(\mathbf{x})\right)\right)$$

here: $\mathbf{y} = \mathbf{W}_{\text{out}} \cdot \varphi\left(\mathbf{W}_{h2} \cdot \varphi\left(\mathbf{W}_{h1}\mathbf{x} + b_{h1}\right) + b_{h2}\right) + b_{\text{out}}$

Note: Activation function is typically left out for last layer

Classifying with MLPs

- For classification, the output of the MLP is *usually* forwarded through a Softmax Function: $sm_i(\mathbf{y}) = \frac{e^{y_i}}{\sum_j e^{y_j}}$
- Softmax can be seen as an additional layer in the MLP
- sm_i is pseudo-probability for class c_i
- Predicted class: $\hat{c} = \max_i sm_i$

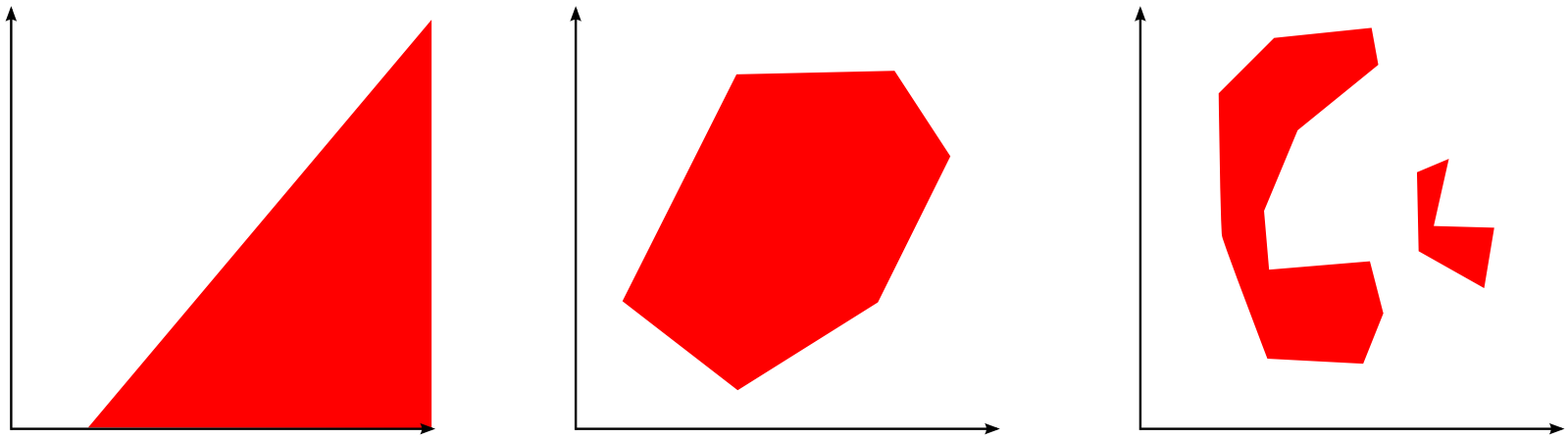


What an MLP Can Do!

... approximate any function (even with only 2 layers!)

[Hornik *et al.* 1989]

Interpretation with 3 layers (2 hidden, 1 output):



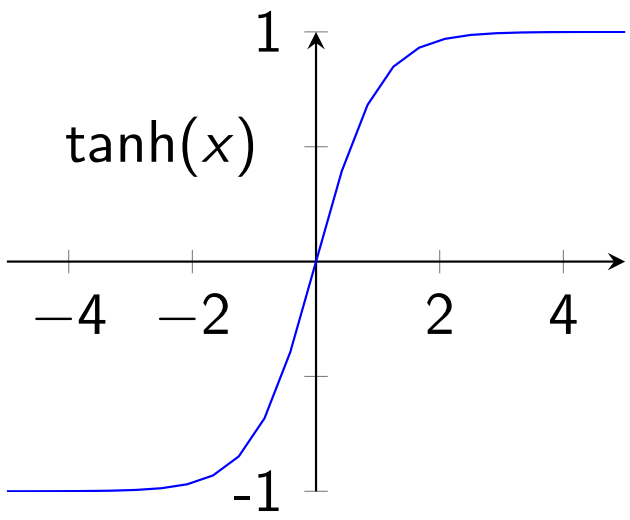
1. Layer: Halfspaces
2. Layer: Convex polyhedron
3. Layer: Multiple non-convex, non-connected polyhedra

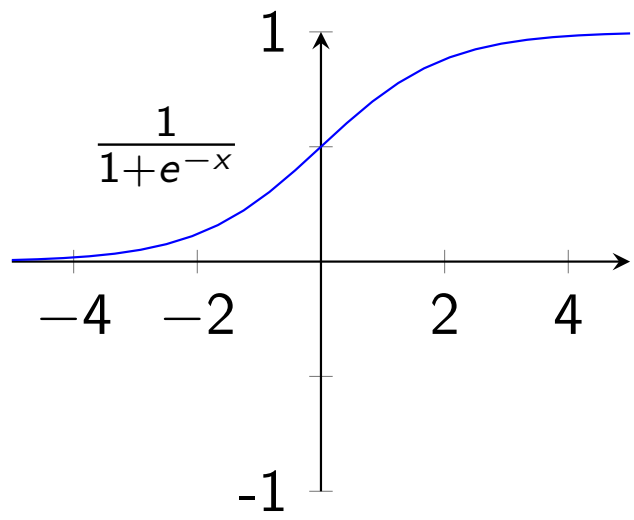
A Word on Activation Functions

- Activation functions are crucial for MLPs
- Without non-linearities, an MLP implements a linear transform:

$$\mathbf{y} = \mathbf{W}_L \mathbf{W}_{L-1} \dots \mathbf{W}_2 \mathbf{W}_1 \mathbf{x} = \mathbf{W}' \mathbf{x}$$

Classic Activation Functions: sigmoidal shape ("threshold-like")





Note: With $\phi_s(x) = \frac{1}{1+e^{(-x)}}$ we have $\frac{\partial \phi_s(x)}{\partial x} = \phi_s(x)\,(1 - \phi_s(x))$.

Training an MLP

How to determine weights such that desired functition is performed?

Basic Idea: Compare (computed) output of MLP

$$\hat{\boldsymbol{y}} = \boldsymbol{f}^L\left(\boldsymbol{f}^{L-1}(\ldots \boldsymbol{f}^2\left(\boldsymbol{f}^1\left(\boldsymbol{x}\right)\right)\right)$$

to *desired* (ideal) output $\boldsymbol{y}$ and update weights such that $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$ become more similar.

Comparison requires *loss function* that evaluates similarity of $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$:

- *Mean Square Error* (MSE):

$$\epsilon_{\text{MSE}} = \frac{1}{2} \cdot \sum_i (y_i - \hat{y}_i)^2$$

- *Cross-Entropy* (in comb. w. Softmax):

$$\epsilon_{\text{CE}} = -\sum_i y_i \log \hat{y}_i$$

Training an MLP II

How to update weights in order to reduce loss?

Compute gradient of loss wrt. the weights:

$$\frac{\partial \epsilon}{\partial w_{ij}^{l}}$$

Update weights in the negative direction of the gradient:

$$w_{ij}^{l} \leftarrow w_{ij}^{l} - \beta \cdot \frac{\partial \epsilon}{\partial w_{ij}^{l}}$$

$\Longrightarrow$ Training the network $\hat{=}$ Gradient Descent

Note: *Learning rate* β controls step size!

Training an MLP: Error Backpropagation

How to compute gradient of loss wrt. network weights?

Define computation of outputs of neurons per layer[1]:

$$f_j^l = \phi(g_j^l) = \phi(\sum_i w_{ij}^l f_i^{l-1})$$

Rewrite gradient applying chain rule:

$$\frac{\partial \epsilon}{\partial w_{ij}^l} = \frac{\partial \epsilon}{\partial f_j^l} \cdot \frac{\partial f_j^l}{\partial g_j^l} \cdot \frac{\partial g_j^l}{\partial w_{ij}^l}$$

$\Rightarrow$ Evaluation *straight forward* for final layer (using ϵ_{MSE} and ϕ_s):

$$\frac{\partial \epsilon}{\partial w_{ij}^L} = -(y_j - f_j^L) \cdot f_j^L(1 - f_j^L) \cdot f_i^{L-1}$$

[1] ignoring bias for siplicity

Training an MLP: Error Backpropagation II

How to evaluate gradient for inner weights?

Hidden layers: define gradient based on "local error" $\delta_j^k = \frac{\partial \epsilon}{\partial g_j^k}$:

$$\frac{\partial \epsilon}{\partial w_{ij}^k} = \frac{\partial \epsilon}{\partial \boldsymbol{f}^L} \cdot \frac{\partial \boldsymbol{f}^L}{\partial \boldsymbol{f}^{L-1}} \cdot \ldots \cdot \frac{\partial \boldsymbol{f}^{k+1}}{\partial \boldsymbol{f}^k} \cdot \frac{\partial \boldsymbol{f}^k}{\partial g_j^k} \cdot \frac{\partial g_j^k}{\partial w_{ij}^k} = \delta_j^k \cdot \frac{\partial g_j^k}{\partial w_{ij}^k}$$

Exploit *generalized chain rule*: For the gradient we obtain:

$$\frac{\partial \epsilon}{\partial f_j^l} = \sum_{k=1} \underbrace{\frac{\partial \epsilon}{\partial f_k^{l+1}} \cdot \frac{\partial f_k^{l+1}}{\partial g_k^{l+1}}}_{\delta_k^{l+1}} \cdot \underbrace{\frac{\partial g_k^{l+1}}{\partial f_j^l}}_{w_{jk}^{l+1}} = \sum_{k=1} w_{jk}^{l+1}\, \delta_k^{l+1}$$

$$\Rightarrow \qquad \delta_j^l = \frac{\partial \epsilon}{\partial f_j^l} \frac{\partial f_j^l}{\partial g_j^l} = \left\{ \sum_{k=1} w_{jk}^{l+1}\, \delta_k^{l+1} \right\} \cdot \frac{\partial f_j^l}{\partial g_j^l}$$

Local error can be computed *backward* through the network

$\Rightarrow$ Error Backpropagation

Minimum-Error and Maximum-Likelihood Training

... can be shown to be equivalent assuming certain distributions of the desired outputs of the network! [Goodfellow *et al.* 2016, Sec. 5.5]

ML-Estimation of (network) parameters $\boldsymbol{\theta}$:

$$\theta_{\text{ML}} = \underset{\boldsymbol{\theta}}{\text{argmax}}\, p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$$

Assuming i.i.d. samples $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_N\}$:

$$\begin{aligned} \theta_{\text{ML}} &= \underset{\boldsymbol{\theta}}{\text{argmax}} \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) \\ &= \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) \end{aligned}$$

Minimum-Error and Maximum-Likelihood Training II

Assuming that network outputs $\boldsymbol{y}$ follow a Gaussian distribution:

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) = \mathcal{N}(y_i|\hat{y}_i(\boldsymbol{x}_i|\boldsymbol{\theta}), \sigma^2)$$

ML-Estimate is then given by:

$$\begin{aligned}
\theta_{\text{ML}} &= \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) \\
&= \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(y_i - \hat{y}_i)^2}{2\sigma^2}\right\} \\
&= \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{i=1}^{N} \left\{-\frac{(y_i - \hat{y}_i)^2}{2\sigma^2}\right\} = \underset{\boldsymbol{\theta}}{\text{argmin}} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \theta_{\text{MSE}}
\end{aligned}$$

Note: Assumed distribution needs to be compatible with activation function in *final* layer!

$\Rightarrow$ here assumed *linear*, i.e., NN performs regression

Minimum-Error and Maximum-Likelihood Training III

Why cast minimum-error optimization into ML framework?

Properties of ML Estimation

- *Consistency*: ML estimate will converge to true θ as $N \to \infty$. (if true distribution lies within the model assumed)
- *Statistical Efficiency*: With increasing N, no other estimator produces lower MSE wrt. true θ.

$\Rightarrow$ ML Estimation often preferred in Machine Learning!

Reminder: For NN training, ML estimation is performed via gradient descent!

Stochastic Gradient Descent (SGD)

Gradient can be computed ...

(1) for complete training data (avg.)

(2) "online" for every new sample.

But *neither* works well!

⇒ Training takes forever!

⇒ Gradient is extremely noisy!

Solution:

Compute estimate of gradient for a small sub-set of the training data (so-called "mini batch")

Note: Samples are drawn *randomly* from the training set!

⇒ Gradient estimate is *stochastic*!

Stochastic Gradient Descent (SGD) II

Disadvantage of mini-batch processing:

Gradient is still somewhat noisy.

Improved Solution:

Introduce so-called *momentum* η $(0 \ll \eta < 1)$:

$$\begin{aligned} w_{ij}^{(t+1)} &\leftarrow w_{ij}^{(t)} + \Delta w_{ij}^{(t+1)} \\ \Delta w_{ij}^{(t+1)} &= \beta(1-\eta)\frac{\partial \epsilon}{\partial w_{ij}^{l}} + \eta \Delta w_{ij}^{(t)} \end{aligned}$$

Note: Computes smoothed sequence of gradient estimates (i.e. sliding average with exponential decay).

Stochastic Gradient Descent (SGD) III

Warning:

In the widely used Caffe toolbox, the gradient is scaled with the actual loss ϵ:

$$w_{ij}^{l} \leftarrow w_{ij}^{l} - \beta \frac{\partial \epsilon}{\partial w_{ij}^{l}} \cdot \epsilon$$

Result: Learning rate is always *dynamic*.

Rationale:

Loss is big/small $\hat{=}$ Optimization is far from/near to solution

$\Rightarrow$ perform big/small steps in gradient direction.

Classifying Images with Neural Networks

Problem:

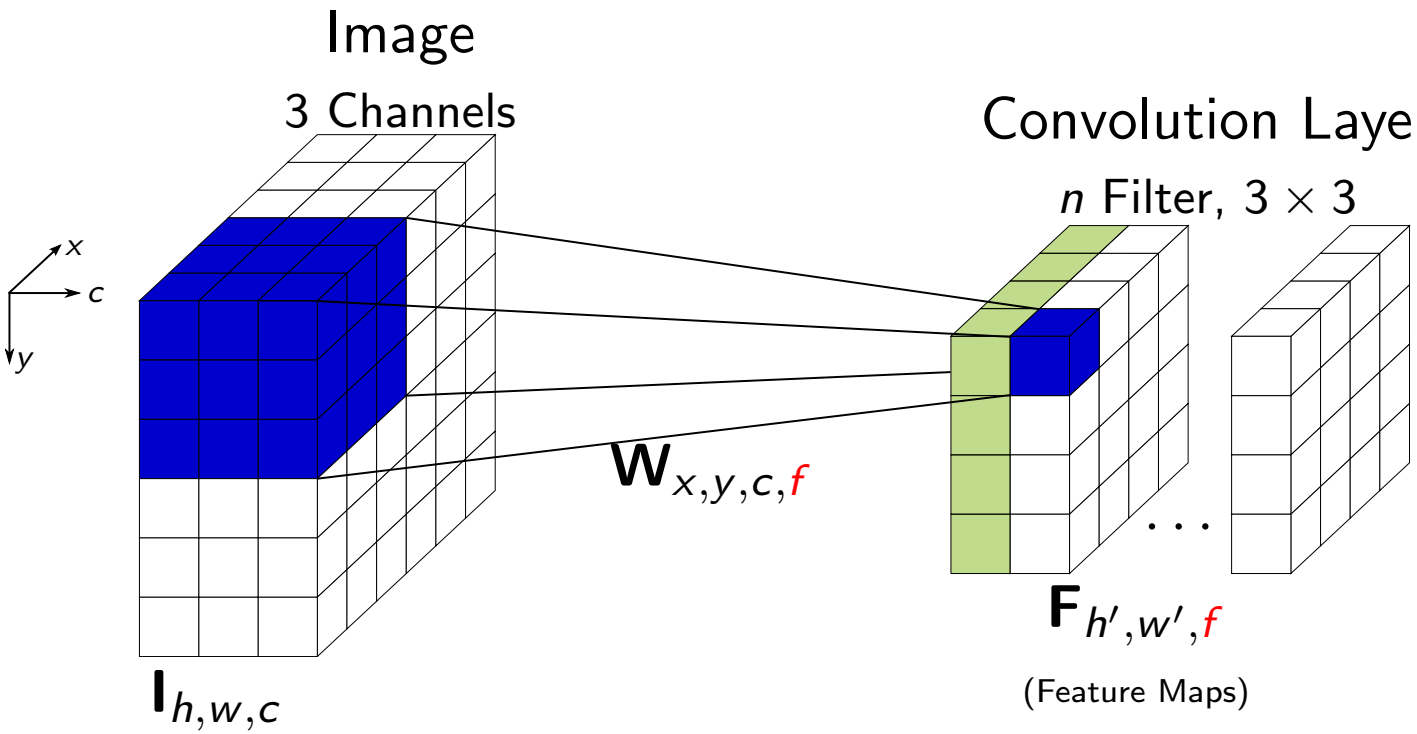
- Using MLPs for image classification is only possible for very small images (e.g. 28×28 pixels)
- Number of weights would explode for bigger images

Example: RGB Image of 224×224 pixels,
first hidden MLP layer has 768 neurons (small layer):
$224 \cdot 224 \cdot 3 \cdot 768 \approx 10^8$ weights in the first layer (441 MB)

Solution: Don't use fully connected layer but rather apply a small number of weights at all possible locations in the image

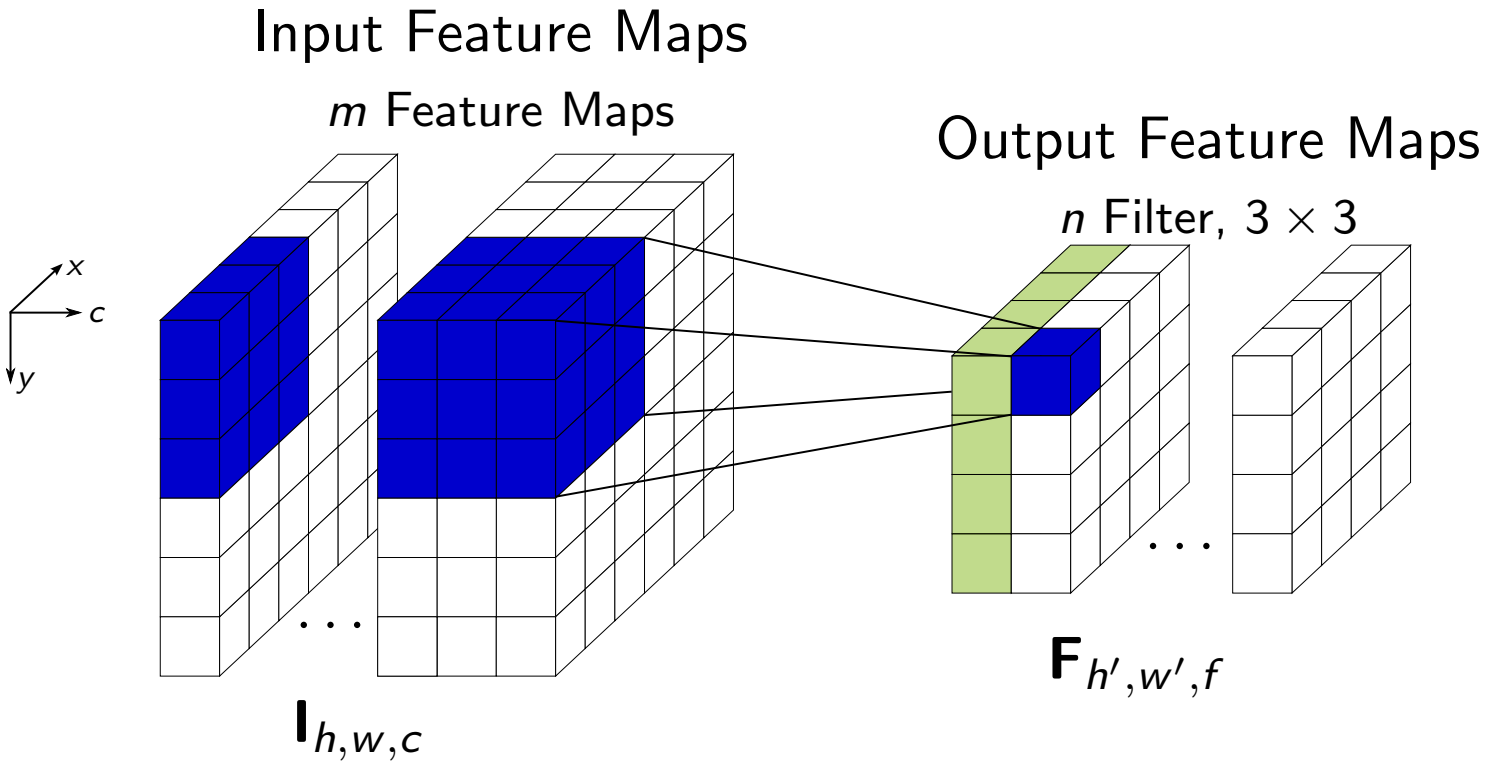
$\Rightarrow$ Convolutional Layer

Convolutional Layer

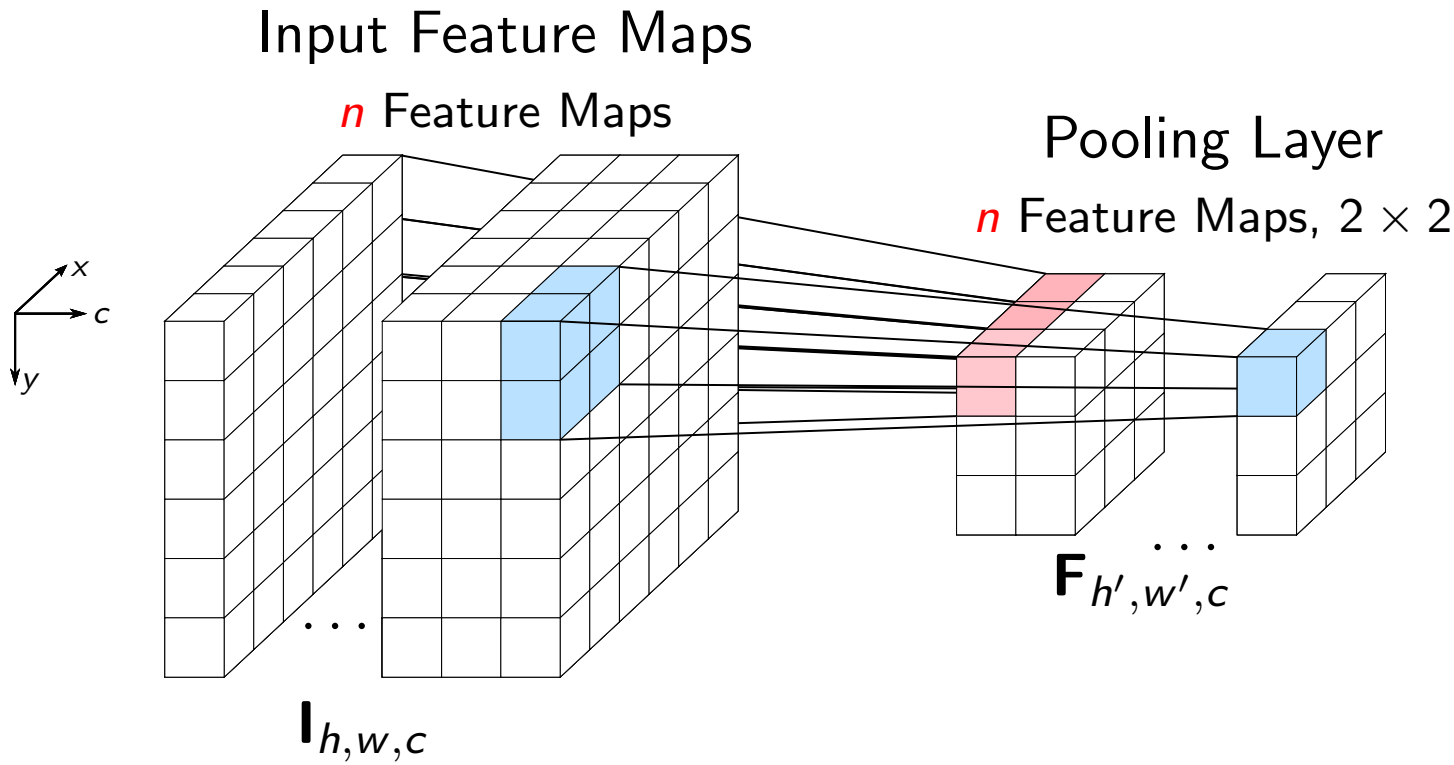


$$\mathbf{F}_{x,y,f} = \varphi\left(\sum_{c=1}^{K}\sum_{i=1}^{3}\sum_{j=1}^{3}\mathbf{W}_{i,j,c,f}\cdot\mathbf{I}_{x+i,y+j,c} + b_f\right)$$

Cascade of Convolutional Layers

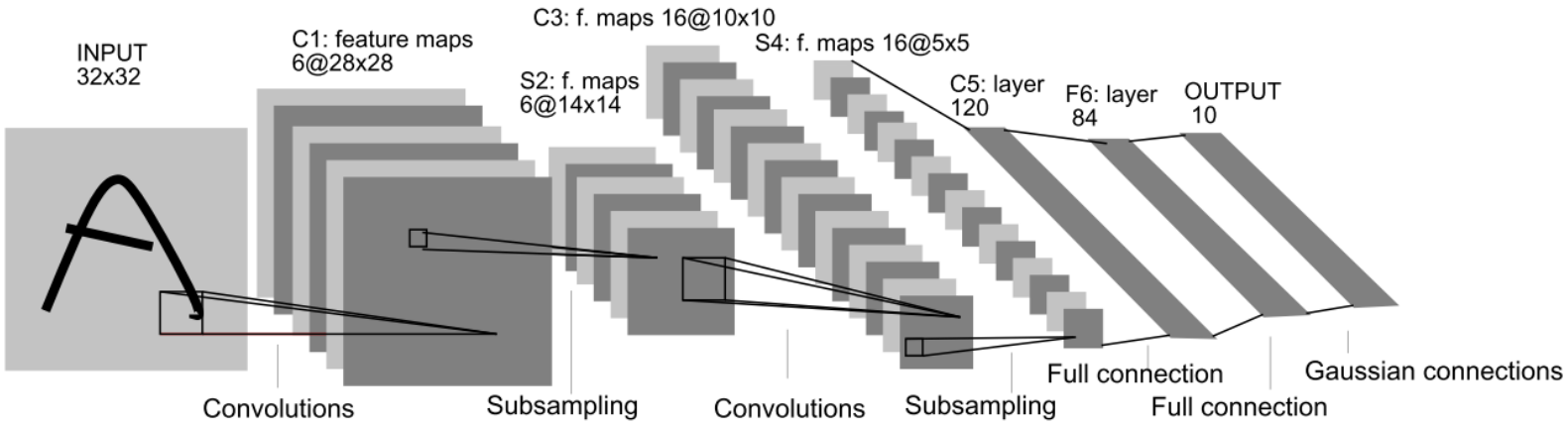


Pooling Layer



$$\mathbf{F}_{x,y,f} = \max_{i,j} \mathbf{I}_{x+i,y+j,f}$$

LeNet



(Source: [LeCun *et al.*, 1990])

- LeNet predicts one of 10 character classes for a given input image
- Subsampling = Pooling Layer
- Gaussian Connections = FC Layer + Euclidean Loss

Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L.D. Jackel: Handwritten Digit Recognition with a Back-Propagation Network, Neural Information Processing Systems, pp. 396–404, 1990.

Deep Learning

In general: Deeper network architectures perform better than shallower ones for vision tasks

Important: Only empirical evidence (no theoretical proofs)

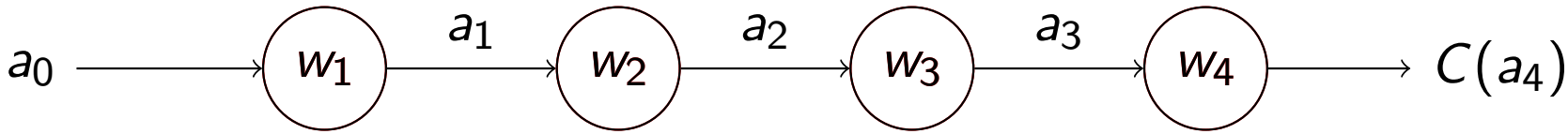
Technically: Deeper means more layers, not a deeper understanding

Even with high computation power and large datasets, Deep Learning did not really pick up until 2012!

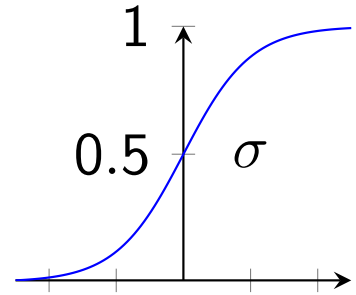
Why? Vanishing Gradient Problem

Vanishing Gradient Problem

Four neuron network, 1D input, 1D output



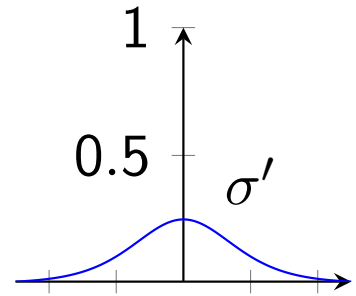
$$z_i = w_i a_{i-1} + b_i \qquad a_i = \sigma(z_i)$$



$$\frac{\partial C}{\partial w_1} = \frac{\partial C}{\partial z_4}\frac{\partial z_4}{\partial a_3} \cdot \frac{\partial a_3}{\partial z_3}\frac{\partial z_3}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_2}\frac{\partial z_2}{\partial a_1} \cdot \frac{\partial a_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

$$= \sigma'(z_4) w_4 \cdot \sigma'(z_3) w_3 \cdot \sigma'(z_2) w_2 \cdot \sigma'(z_1) a_0$$

$$= \sigma'(z_4)\sigma'(z_3)\sigma'(z_2)\sigma'(z_1) \cdot w_4 w_3 w_2 a_0$$



$$= \underbrace{\sigma'(z_4)\sigma'(z_3)\sigma'(z_2)\sigma'(z_1)}_{\leq \frac{1}{4}\cdot\frac{1}{4}\cdot\frac{1}{4}\cdot\frac{1}{4}} \cdot w_4 w_3 w_2 a_0$$

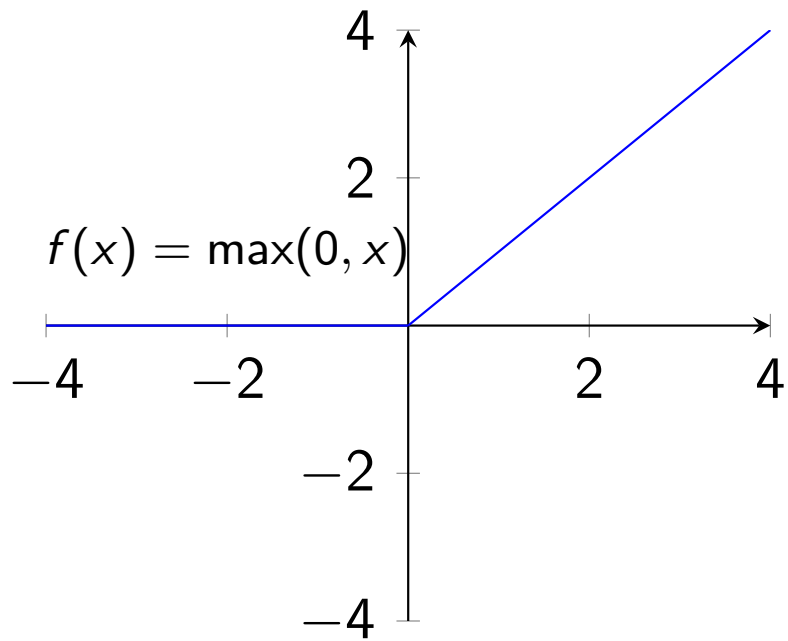
$$\leq \frac{1}{256} \cdot w_4 w_3 w_2 a_0$$

Vanishing Gradient Problem

- Derivative of sigmoidal activation functions < 1
- Exponential decay of gradient magnitude

Desirable: Activation function with derivative $= 1$ but non-linear ($> 1 =$ exploding gradient)

Solution: Rectified Linear Unit (ReLU) [Glorot & Bengio 2010]



How to Get Along With Limited Training Data?

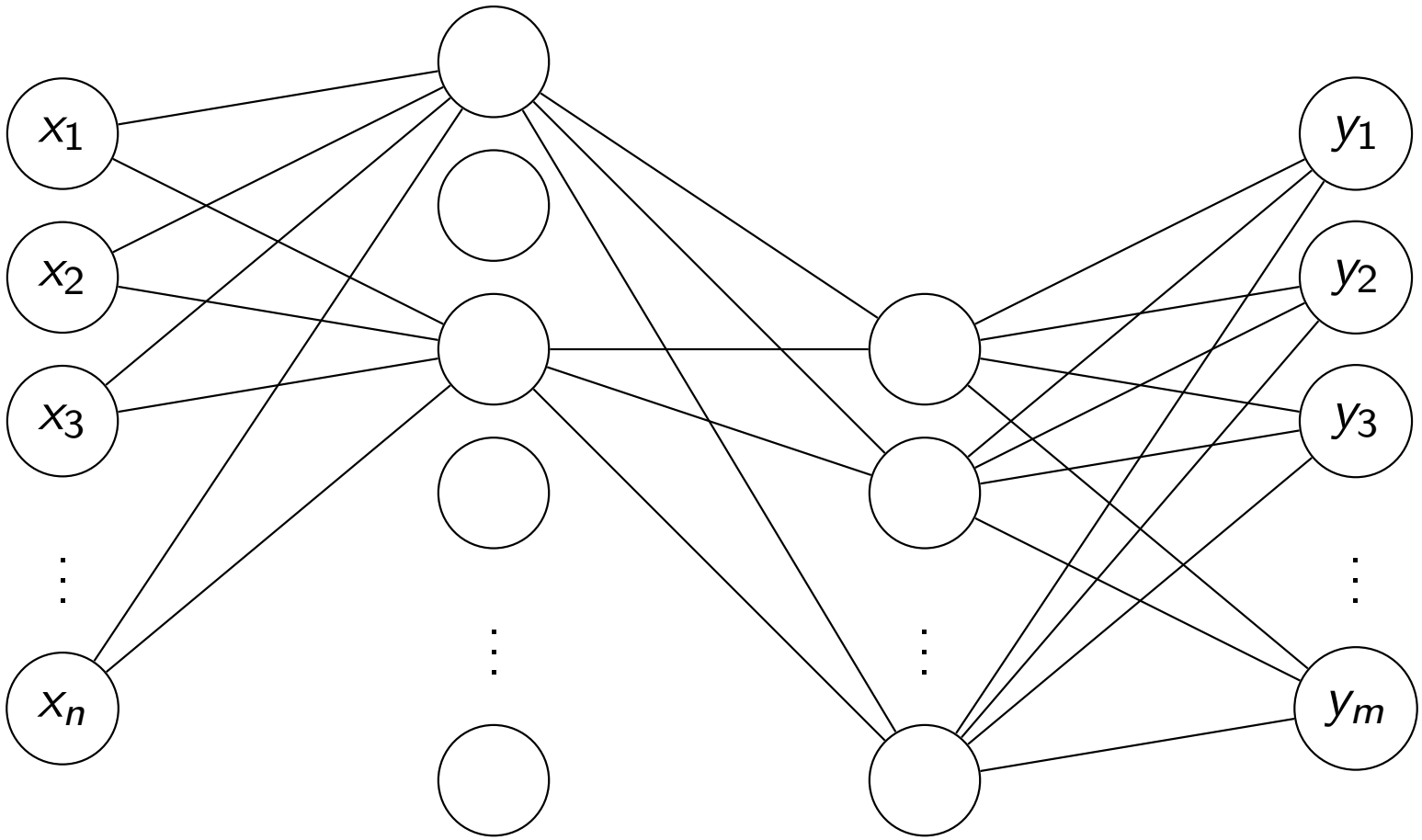
Problem: CNNs easily contain billions of parameters (weights)!
$\Rightarrow$ Could easily learn training samples "by heart".

Solution: Apply *Regularization* during training

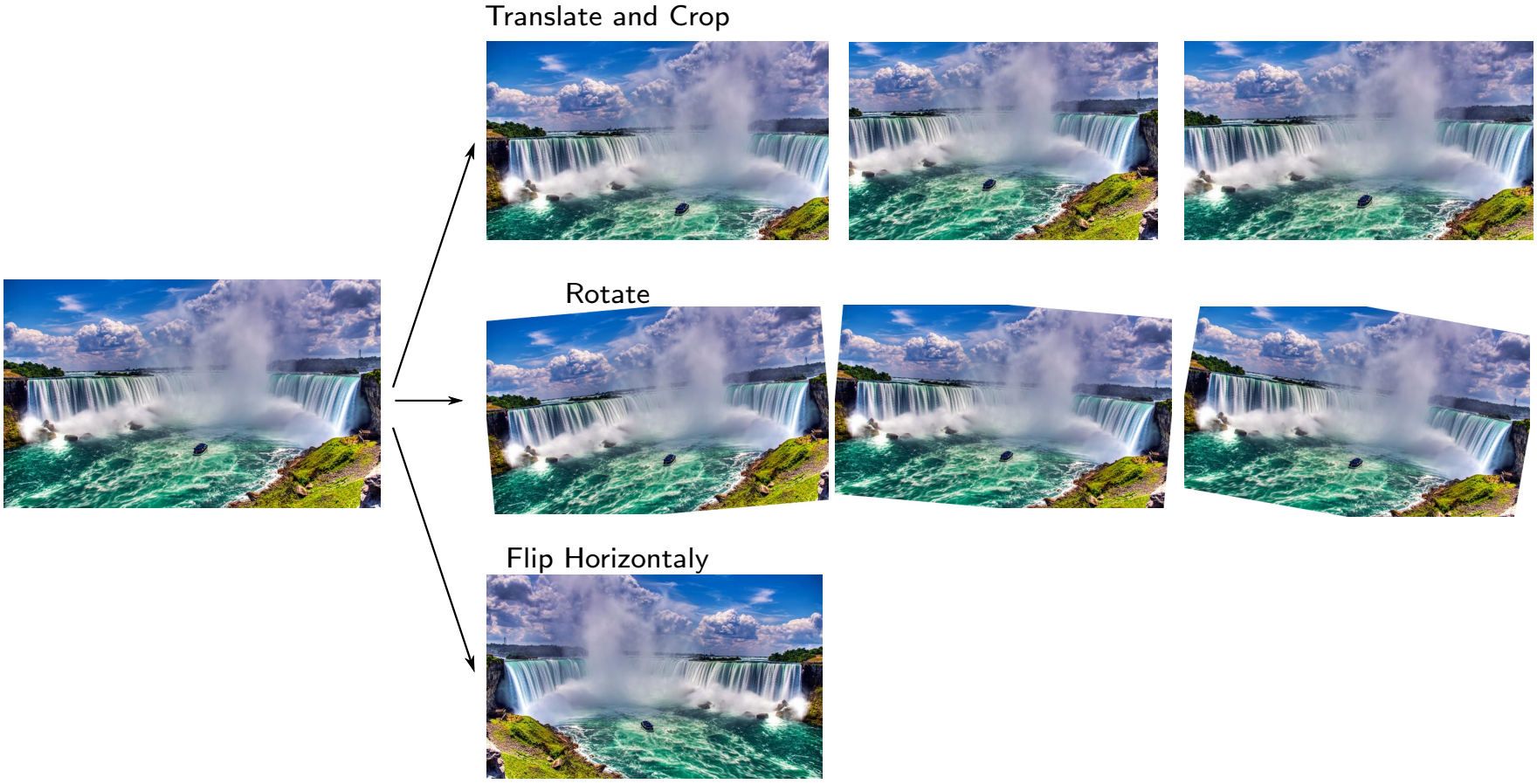
Fundamental Techniques:

- *Convolutional layers*
- *Dropout*
  Randomly set outputs of neurons to zero
  (usually 50% of fully connected layers)
- *Data Agumentation:*
  Generate new, slightly different training samples from existing ones by certain transforms
  (e.g. slight translations, rotations, ...)

Dropout

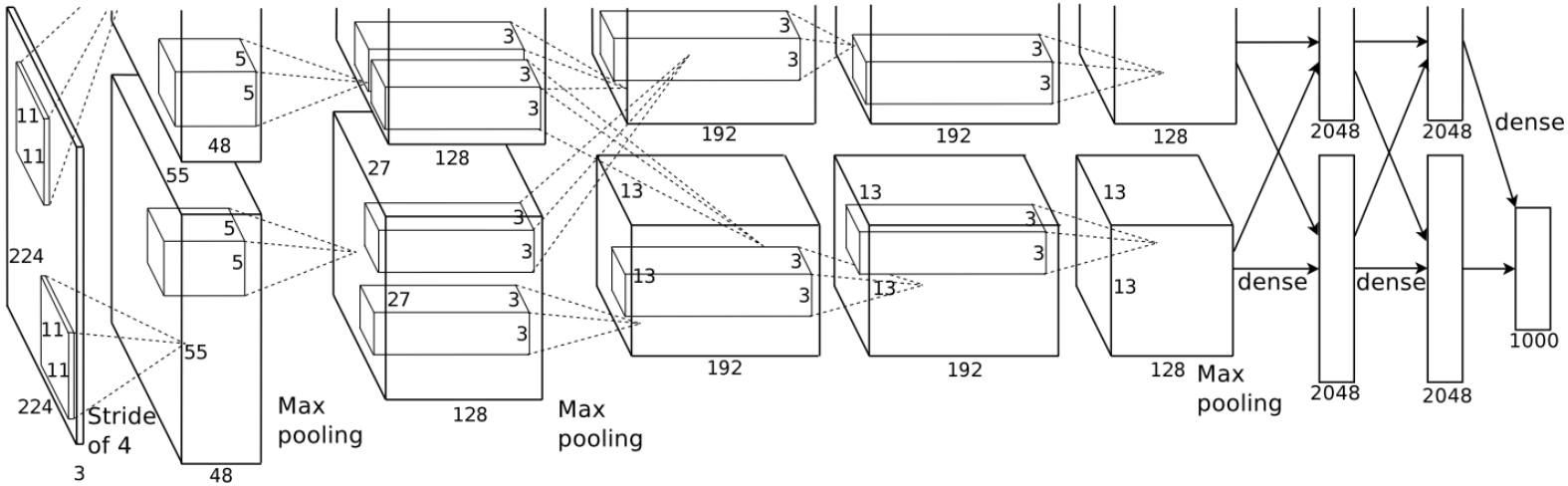


Data Augmentation



Note: Usually different augmentation techniques are mixed to create a single augmented image

Well-known Deep Learning Architectures: AlexNet

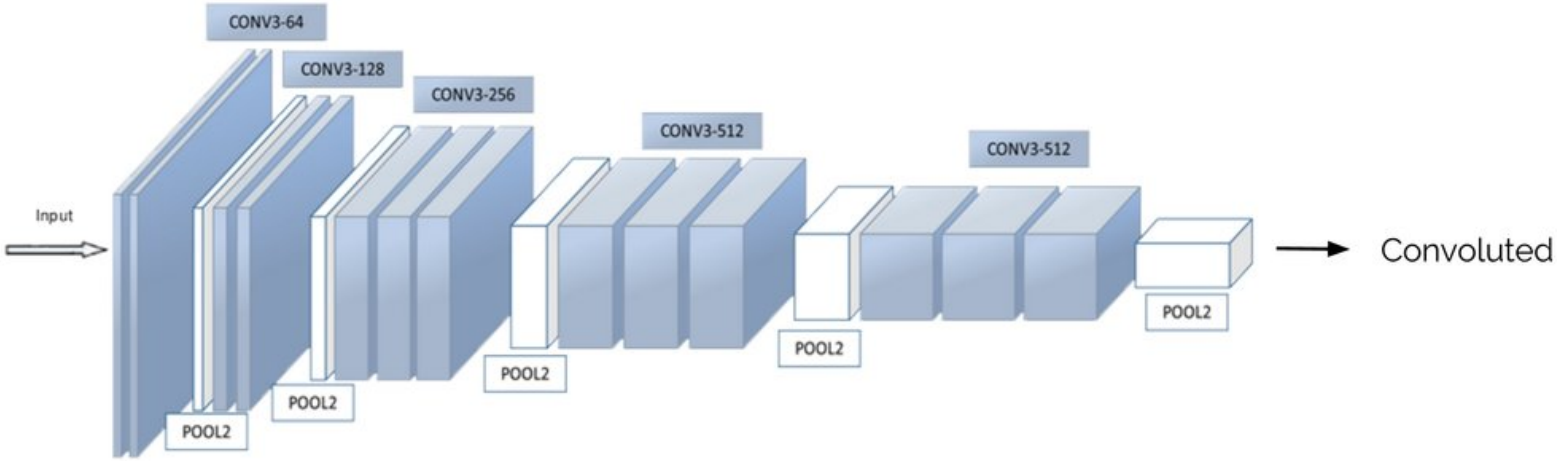


(Source: [Krizhevsky *et al.*, 2012])

- CNN which kicked off the current Deep Learning hype
- Architecture similar to LeNet but more layers/parameters
- Trained on two graphic cards for over a week on ImageNet

A. Krizhevsky, I. Sutskever, G. E. Hinton: ImageNet Classification with Deep Convolutional Neural Networks, Neural Information Processing Systems, pp. 1097–1105, 2012.

Well-known Deep Learning Architectures: VGGNet



(Source: http://html.scirp.org/)

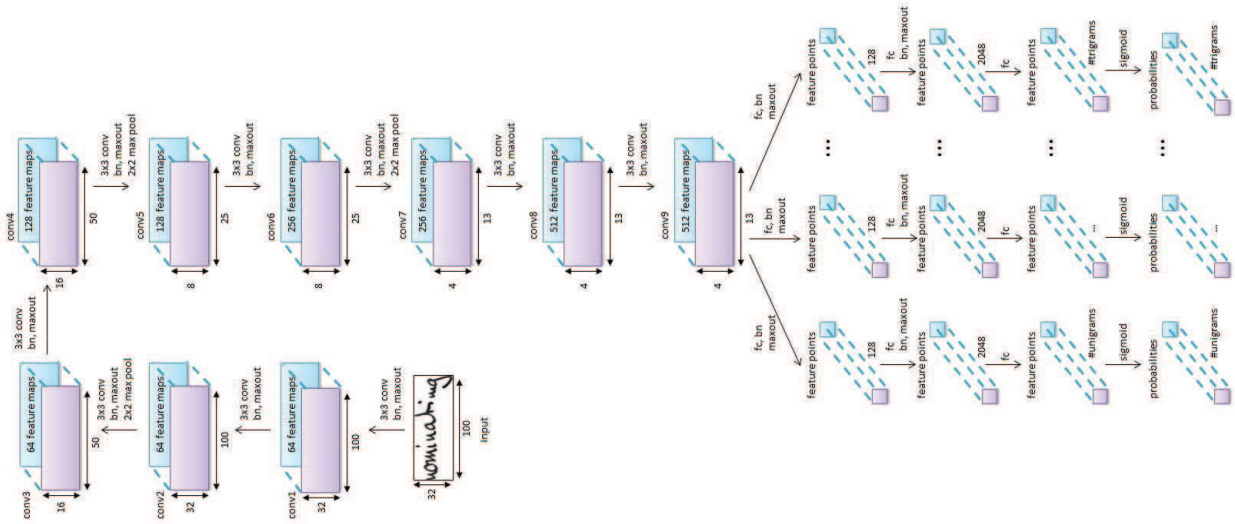
- First CNN to use only 3×3 convolutions (standard for current CNNs)
- Low number of filters in the early layers, high number of filters in the later layers
- Anytime pooling is applied, the number of filters is doubled

K. Simonyan, A. Zisserman: Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv, 2014.

Overview

Related Work on Deep Learning

CNN-N-Gram [Poznanski *et al.*, 2016]

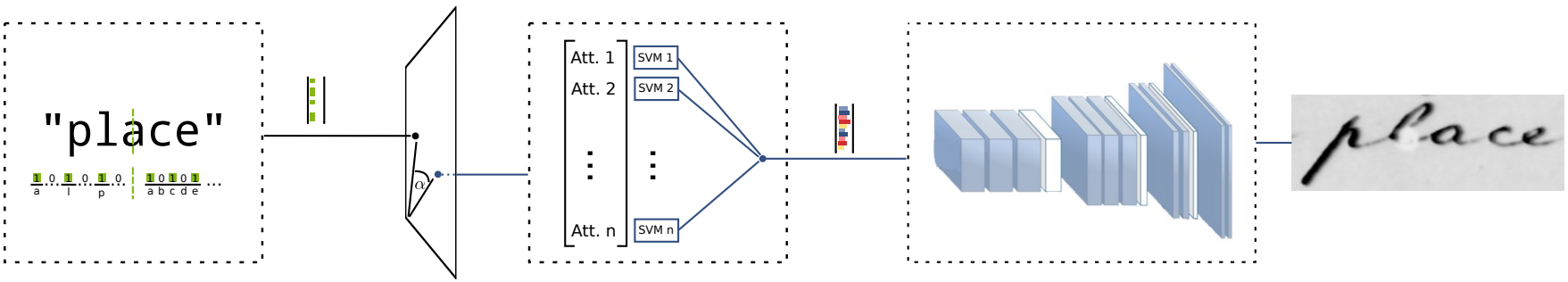


- Considers isolated word *recognition*
- Uses PHOC + 1 level of trigrams
- Attributes are predicted directly with *separate* MLPs

Poznanski, A., Wolf, L.: CNN-N-Gram for Handwriting Word Recognition, IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition, pp. 2305–2314, Las Vegas, USA, 2016.

Related Work on Deep Learning for Word Spotting

Deep Feature Embedding [Krishnan *et al.*, 2016]

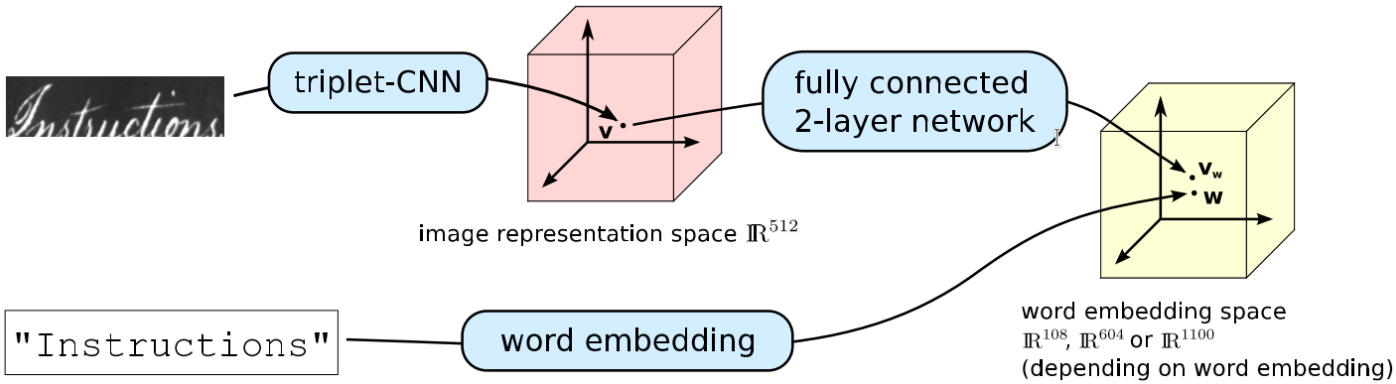


- Uses CNN to produce feature representation of images
- CNN is pre-trained on synthetically generated data
- SVMs predict attributes of PHOC representation

Krishnan, P., Dutta, K., Jawahar, C. V.: Deep Feature Embedding for Accurate Recognition and Retrieval of Handwritten Text, Int. Conf. on Frontiers in Handwriting Recognition, Shenzhen, China, pp. 289–294, 2016.

Related Work on Deep Learning for Word Spotting II

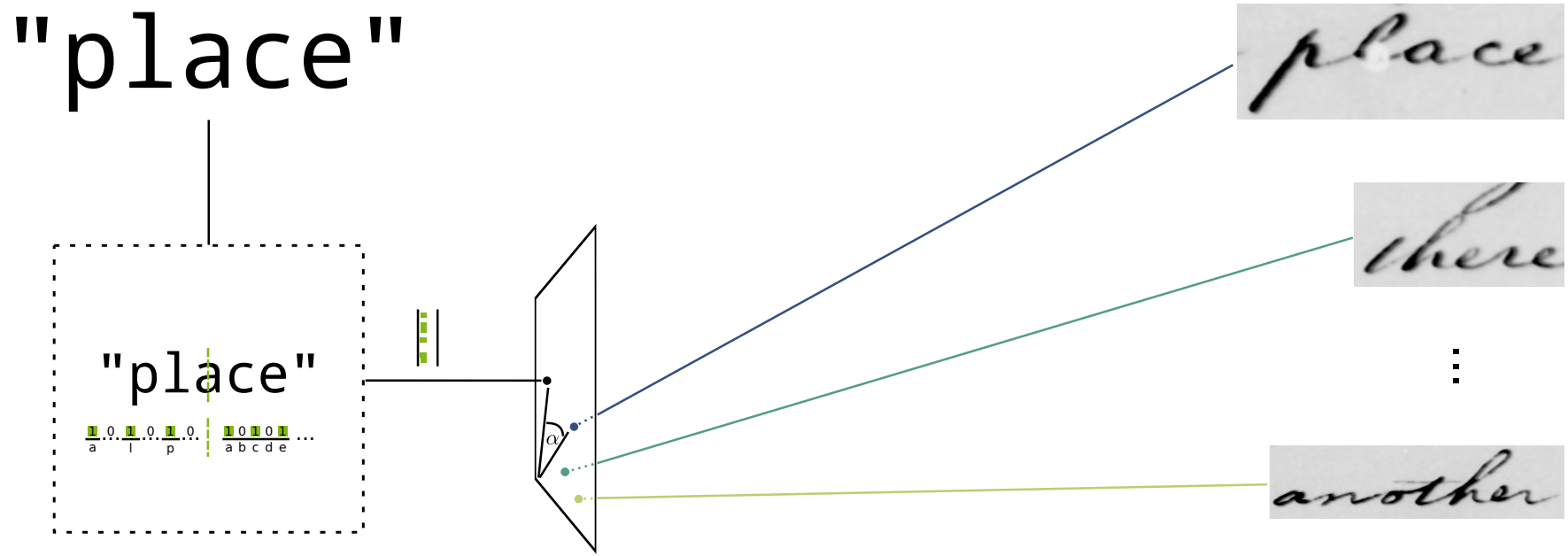
Triplet-CNN [Wilkinson & Brun, 2016]



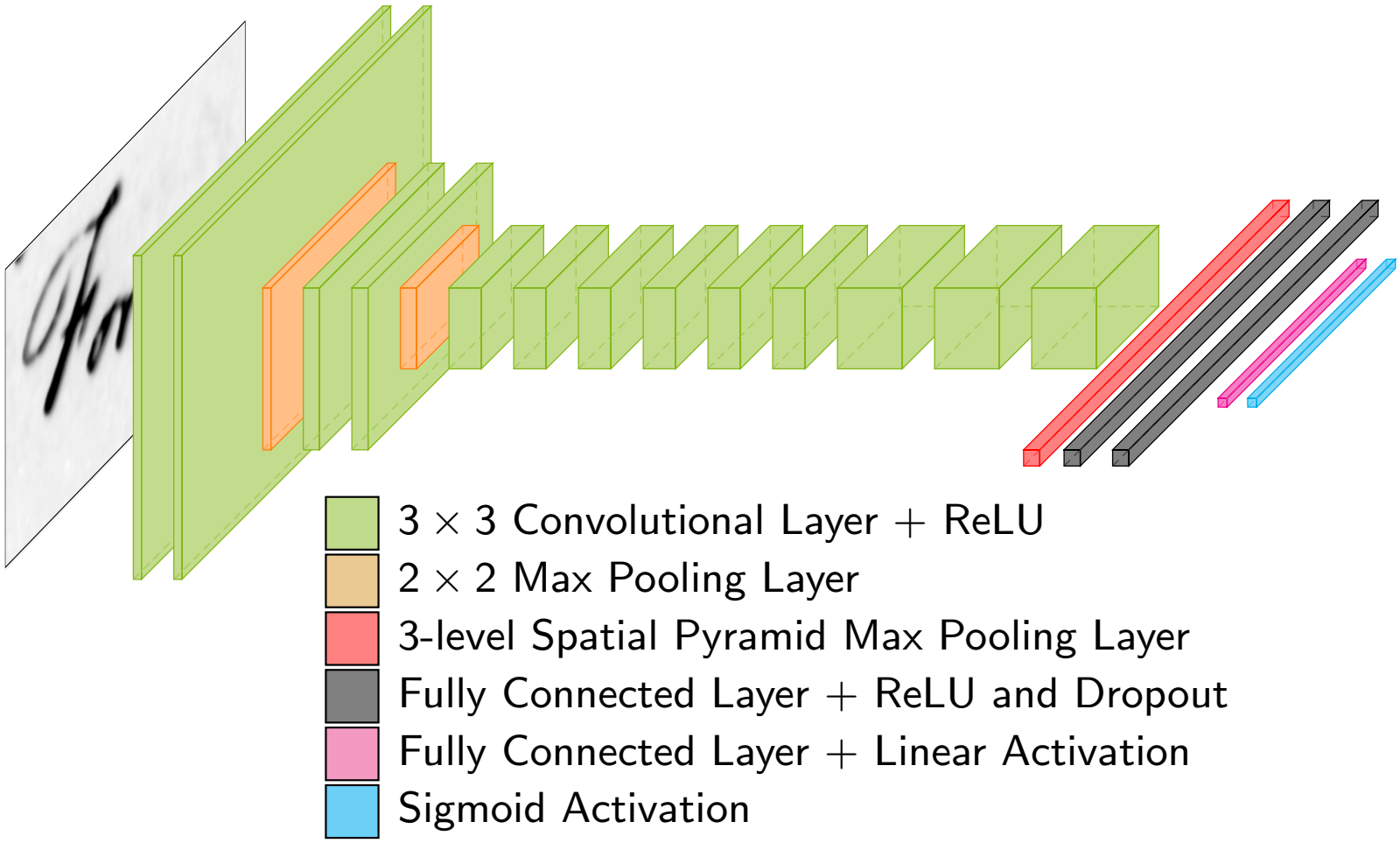
- Residual network learns word descriptors using triplet loss
- Separate MLP predicts attribute representation
- Proposed *DCT of Words* embedding

Wilkinson, T., Brun, A.: Semantic and verbatim word spotting using deep neural networks. Int. Conf. on Frontiers in Handwriting Recognition, Shenzhen, China, pp. 307–312, 2016.

Reminder: General Framework



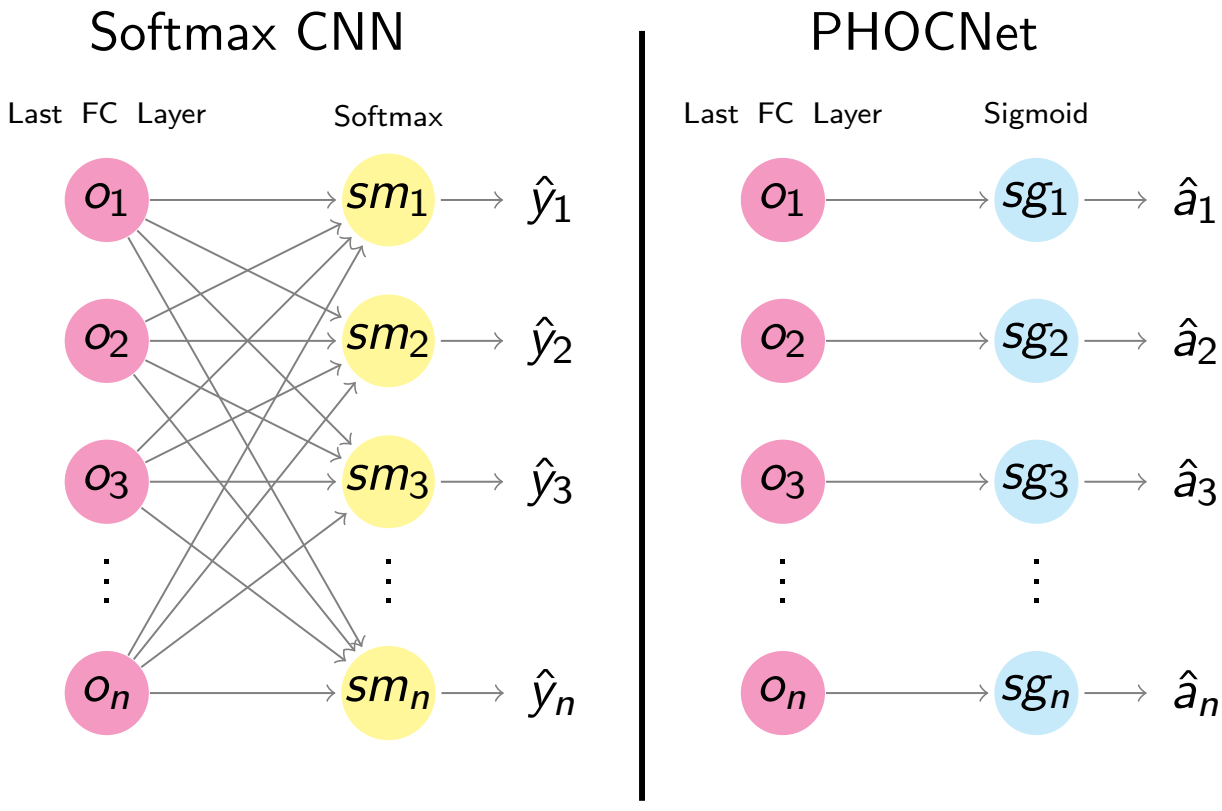
PHOCNet



S. Sudholt, G. A. Fink: PHOCNet: A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents, Proc. Int. Conf. on Frontiers in Handwriting Recognition, Shenzhen, China, 2016.

Softmax CNN vs. PHOCNet

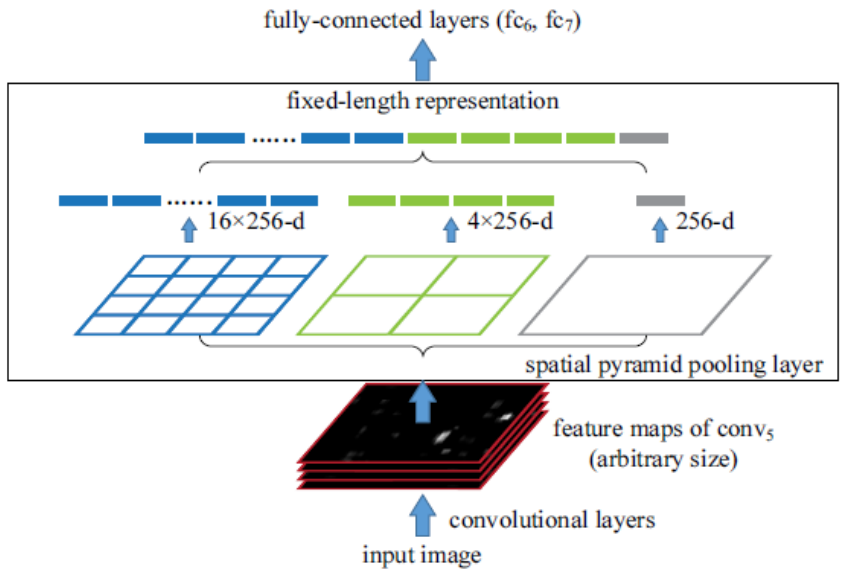
- In order to classify attributes, replace softmax with a sigmoid activation
- Each output neuron predicts one attribute



Spatial Pyramid Pooling Layer

- Convolutional layers can already deal with arbitrary image sizes
- Only MLP part has a problem with changing image sizes

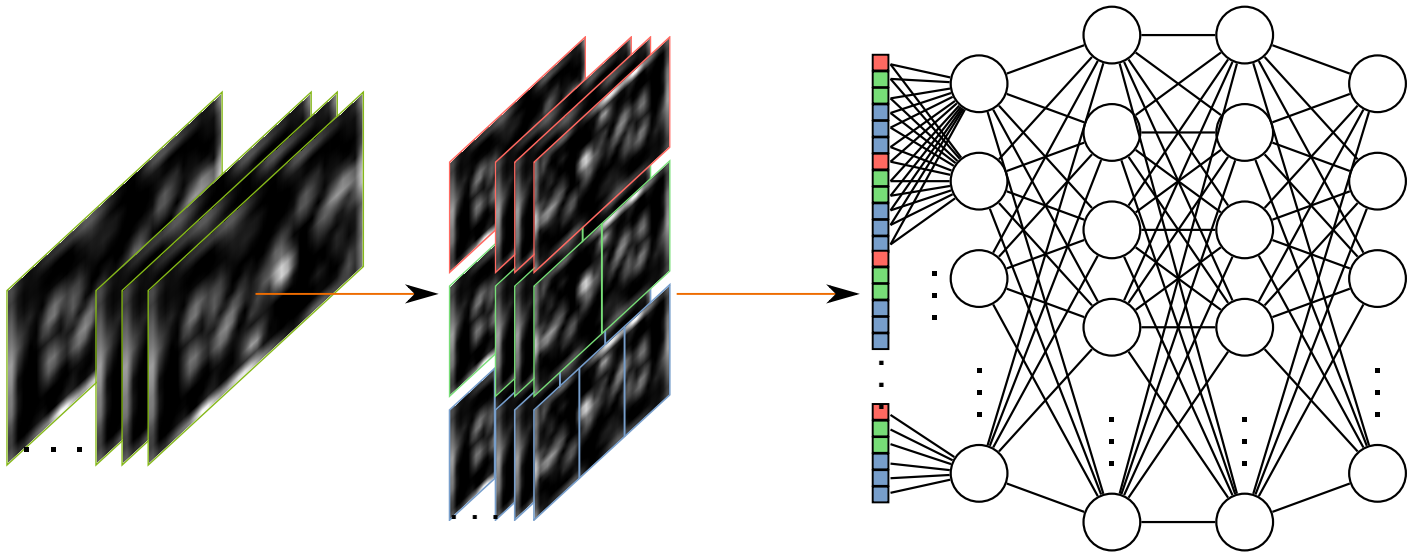
Solution: apply spatial pyramid concept to the last convolutional output to generate fixed-size representation



K. He, X. Zhang, S. Ren, J. Sun: Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, Proc. European Conference on Computer Vision, pp. 346–361, 2014.

Temporal Pyramid Pooling Layer

- Pooling focusses on *horizontal* axis, i.e., writing direction, only!



- Input: All k feature maps from last convolutional layer
- Performs pyramidal pooling along horizontal axis for each feature map and with different splits (cf. PHOC)
- Produces fixed size input for MLP

Sudholt, S., Fink, G. A.: Evaluating Word String Embeddings and Loss Functions for CNN-based Word Spotting, Proc. Int. Conf. on Document Analysis and Recognition, Kyoto, Japan, 2017.

Training the PHOCNet: Loss Functions

Reminder: Casting NN optimization into ML framework allows to derive loss functions.

Procedure:
1. Define (assumed) distribution of label data
2. Set up (negative) log-likelihood function and derive loss

What are appropriate distribution assumptions for PHOC vectors?

- Every attribute can be considered a binary variable, i.e., with Bernoulli distribution

$$p_{\mathcal{B}}(k|p) = p^k(1-p)^{(1-k)} \quad \text{for } k \in \{0,1\}$$

(here: p corresponds to "success" probability, $k = 1$)

- Attributes exhibit dependencies but modeling these is prohibitive!
  $\Rightarrow$ assume pair-wise independent PHOC attributes

Training the PHOCNet: Loss Functions II

Minimize negative log-likelihood for vector of D pair-wise independent Bernoulli-distributed attributes:

$$\begin{aligned}
\boldsymbol{\theta}_{\text{ML}} &= \underset{\boldsymbol{\theta}}{\text{argmin}} - \log \prod_{i=1}^{N} \prod_{d=1}^{D} p_{\mathcal{B}}(y_i^{(d)} | \hat{y}_i^{(d)}(\boldsymbol{x}_i | \boldsymbol{\theta})) \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \sum_{d=1}^{D} \log p_{\mathcal{B}}(y_i^{(d)} | \hat{y}_i^{(d)}) \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \sum_{d=1}^{D} \log \left\{ (\hat{y}_i^{(d)})^{y_i^{(d)}} \cdot (1 - \hat{y}_i^{(d)})^{(1-y_i^{(d)})} \right\} \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \sum_{d=1}^{D} y_i^{(d)} \log \hat{y}_i^{(d)} + (1 - y_i^{(d)}) \log(1 - \hat{y}_i^{(d)})
\end{aligned}$$

Note: Usually referred to as *Binary Cross Entropy Loss* or *Sigmoid Cross Entropy Loss*

Training the PHOCNet: Loss Functions III

Alternative view on a loss for PHOC representations:

- Euclidean distance / MSE loss surely not suitable!

Reason: Curse of dimensionalty, i.e., Euclidean distance becomes meaningless in high-dimensional spaces!

How can pair-wise independence assumtion be avoided?

$\Rightarrow$ Consider binary vectors as a whole!

Observation: Cosine dissimilarity works well for *directional* data and likewise for *histogram-like* data!

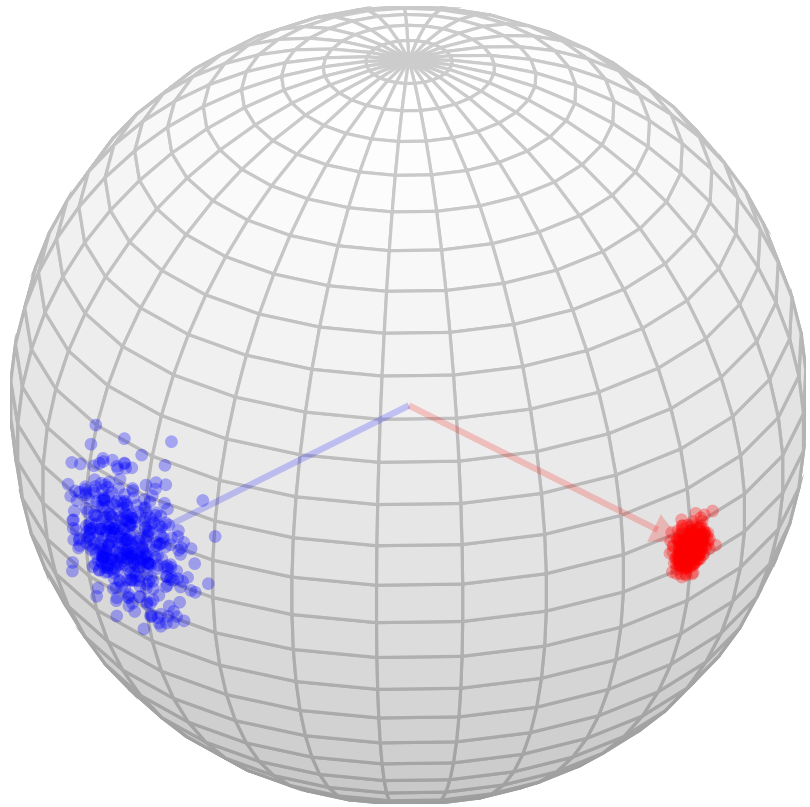
$$d_{\cos}(\boldsymbol{a}, \boldsymbol{b}) = 1 - \frac{\boldsymbol{a}^T \boldsymbol{b}}{||\boldsymbol{a}|| \cdot ||\boldsymbol{b}||}$$

Reason: Not the length of the vector but the direction matters! (direction $\approx$ shape of the histogram)

Training the PHOCNet: Loss Functions IV

Distribution for directional data: Von Mises-Fisher distribution

- Normal distribution on the unit hypersphere
- Parameters: *mean direction* $\boldsymbol{\mu}$ and *concentration* κ ($\approx$ inverse variance)



$$p_{\mathcal{MF}}(\boldsymbol{x}|\boldsymbol{\mu},\kappa) = C_D(\kappa)\exp(\kappa\,\boldsymbol{\mu}^T\boldsymbol{x})$$

with $||\boldsymbol{\mu}|| = ||\boldsymbol{x}|| = 1$
and $C_D(\kappa)$ a normalization factor

Training the PHOCNet: Loss Functions IV

Minimize negative log-likelihood for vectors following von-Mises-Fisher distributions (with identical κ):

$$
\begin{aligned}
\boldsymbol{\theta}_{\text{ML}} &= \underset{\boldsymbol{\theta}}{\text{argmin}} - \log \prod_{i=1}^{N} p_{\mathcal{MF}}(\boldsymbol{y}_i | \hat{\boldsymbol{y}}_i(\boldsymbol{x}_i | \boldsymbol{\theta}), \kappa) \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \log p_{\mathcal{MF}}(\boldsymbol{y}_i | \hat{\boldsymbol{y}}_i) \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \log C_D(\kappa) \exp(\kappa \, \hat{\boldsymbol{y}}_i^T \boldsymbol{x}_i) \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} - \sum_{i=1}^{N} \kappa \, \hat{\boldsymbol{y}}_i^T \boldsymbol{x}_i = \underset{\boldsymbol{\theta}}{\text{argmin}} \sum_{i=1}^{N} 1 - \hat{\boldsymbol{y}}_i^T \boldsymbol{x}_i \\
&= \underset{\boldsymbol{\theta}}{\text{argmin}} \sum_{i=1}^{N} 1 - \frac{\hat{\boldsymbol{y}}_i^T \boldsymbol{x}_i}{||\hat{\boldsymbol{y}}|| \cdot ||\boldsymbol{x}_i||} = \underset{\boldsymbol{\theta}}{\text{argmin}} \sum_{i=1}^{N} d_{\cos}(\hat{\boldsymbol{y}}_i, \boldsymbol{x}_i)
\end{aligned}
$$

Training the PHOCNet: Weight Initialization

Problem: SGD optimization of network parameters needs initial parameter set $\boldsymbol{\theta}_0$ to start from!

Observation: Training of deep networks is quite sensitive to the choice of this initialization!

Procedure for the PHOCnet: [He *et al.* 2015]

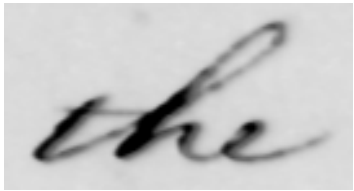
$$w_{ij} \sim \mathcal{N}\left(0, \frac{2}{n_j}\right) \quad \text{and} \quad b_j \leftarrow 0$$

(Here n_j is equal to the number of inputs per neuron.)

Note: Referred to as 'MSRA' (Microsoft Research Asia) in *Caffe* and `kaiming_normal_(...)` in *Pytorch*.

Word Spotting: Evaluation

Query word image



or string ("the")

Retrieved patches sorted by score

- *Precision*: How relevant is the list?
- *Recall*: How complete is the list w.r.t. relevant items?
- *Average Precision*: How well is the retrieval list sorted?
  (Implicitly takes Recall into account!)

Notes:

- Usually mean values over many queries are reported (mAP and mR).
- Patch overlap threshold required for segmentation-free case.
- Transform result into relevant / ...

Word Spotting: Evaluation II

Average Precision: How well is the retrieval list sorted?

- Let's make the example a little more complex:

$$[1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1]$$

  Total number of relevant items in dataset: 10
- Precision: $\frac{8}{15} \approx 0.53$, Recall: $\frac{8}{10} = 0.8$
  No information about the list's order!
- Average Precision: Precision averaged at different recall levels (cf. e.g. [Lladós *et al.* 2012]):

$$\frac{\sum_{k=1}^{n} \text{Precision}_k \times \text{rel}(k)}{\#\text{Relevant Items in Dataset}}$$

  $\text{rel}(k)$: Relvance of item k, Precision_k: Precision at cut-off k

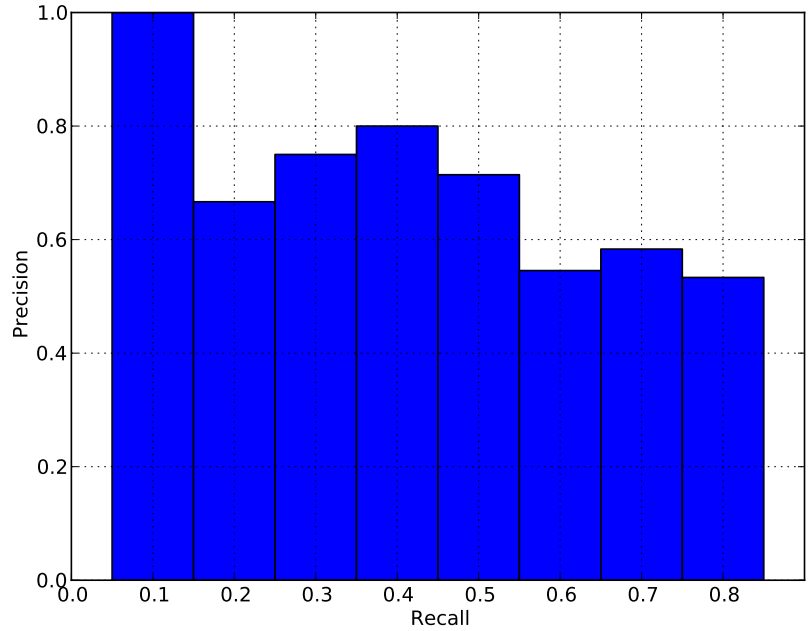
  Accumulate Precision whenever the Recall changes and normalize.

Word Spotting: Evaluation III

$$[1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1]$$

Average Precision:

$$\frac{\sum_{k=1}^{n} \text{Precision}_k \times \text{rel}(k)}{\#\text{Relevant Items in Dataset}}$$



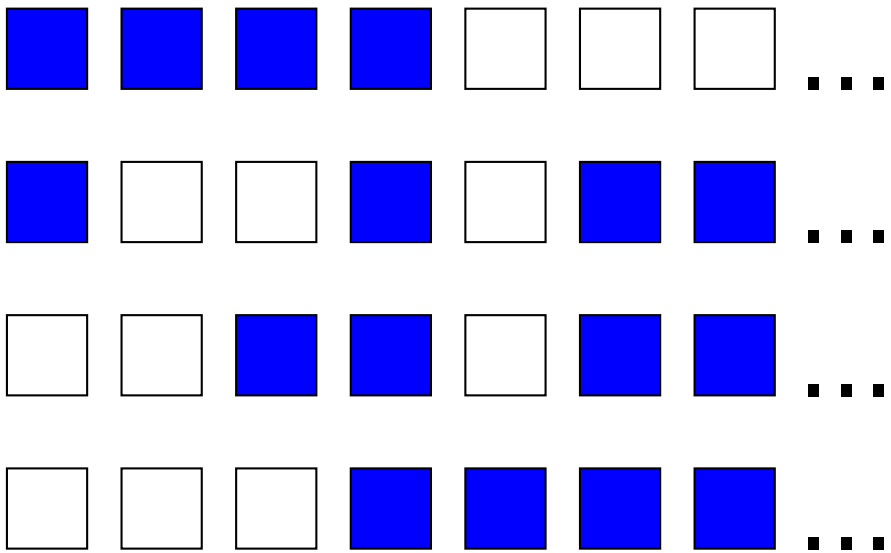
$$\frac{\frac{1}{1} + \frac{2}{3} + \frac{3}{4} + \frac{4}{5} + \frac{5}{7} + \frac{6}{11} + \frac{7}{12} + \frac{8}{15}}{10}$$

$$\approx 0.56$$

Word Spotting: Evaluation IV

Mean Average Precision: Average of AP for different queries

Example (with 4 relevant items to retrieve):



AP: 1.00

AP: 0.64

AP: 0.48

AP: 0.43

mAP: 0.63

Evaluation: Significance Testing

Question: Is a mAP of 0.93 saying that a method is better than another with a mAP of 0.91/0.77/0.3?

Problem: *You never know! Could be random effects!*

Solution: Test difference of evaluation results for *statistical significance*!

Method of choice: *Permutation Test*
(requires no assumptions about distribution of test statistic)

Evaluation: Permutation Test

... also known as randomization test (cf. Good 2000)

Null Hypothesis: Samples A and B obtained from two different sources (here: results of word spotting methods) follow the same underlying distribution, (i.e., systems perform the same).

Basic idea: Reject null hypothesis if difference T_{obs} between sample means μ_A and μ_B (here: mAP) is large enough.

Procedure: Generate all possible N permutations of assigning samples to sets A and B and compute difference T_n of means.

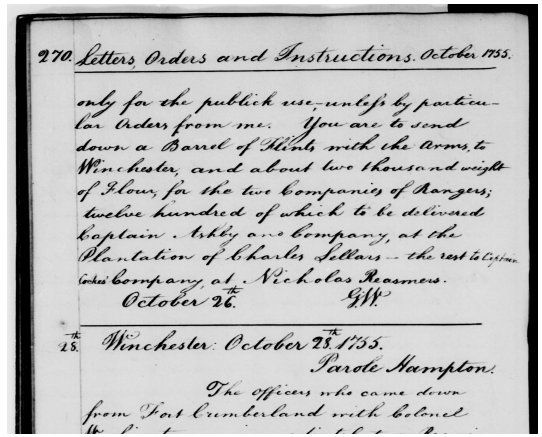
Result: Proportion of differences $T_n \geq T_{\text{obs}}$ is Probability (p-value) of accepting the null hypothesis.

$\Rightarrow$ For small p it is concluded that sets A and B *do not follow the same distribution* (systems perform differently)!

Evaluation: Data Sets I

George Washington Benchmark:
Database of handwritten letters

- Likely single-writer documents
- 20 pages, 4860 words
- 4-fold cross validation



Esposalles Benchmark: Database of marrige license books

- High script variability
- Degradation (e.g. bleed through)
- 32k training / 13k test word images

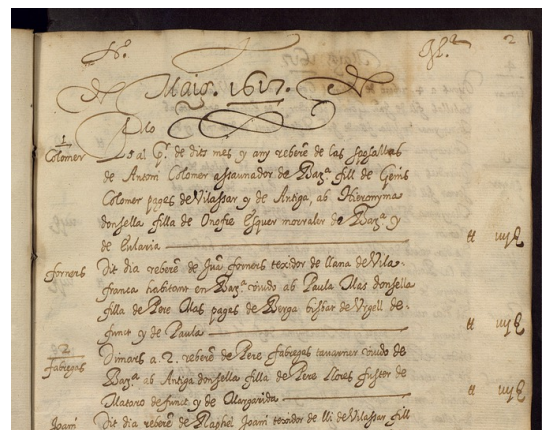
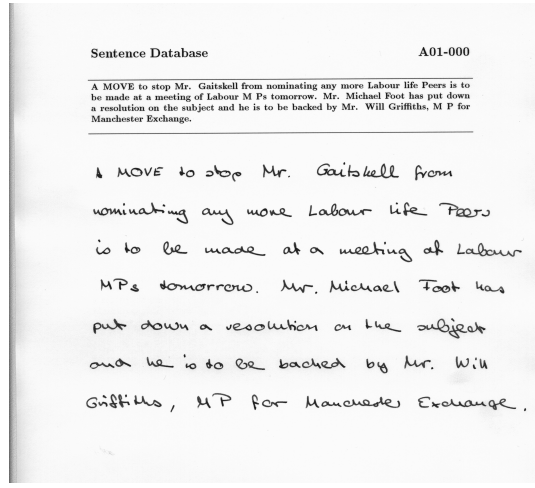


Image sources:
The George Washington Papers at the Library of Congress, 1741-1799
V. Romero *et al.*: *The ESPOSALLES Database: An Ancient Marriage License Corpus for Off-line Handwriting Recognition*, Pattern Recognition, Volume 46(6), pp. 1658-1669, 2013.

Evaluation: Data Sets II

IAM Database: Handwritten paragraphs (elicited)

- more than 600 writers
- more than 13k lines / 115k words
- 6k/1.8k/1.8k train/val/test lines



IFN/ENIT Database: Handwritten Tunesian city names (elicited, Arabic)

- more than 400 writers / 26k words
- training (A, B, C); test (D)
- reduced Arabic character set (50)

Image sources:
U. V. Marti, H. Bunke: *The IAM-Database: An English Sentence Database for Offline Handwriting Recognition*, IJDAR, vol. 5, pp. 39-46, 2002.
M. Pechwitz, S. Snoussi Maddouri, V. Märgner, N. Ellouze, H. Amiri: *IFN/ENIT-Database of handwritten Arabic words*, Proc. 7th Colloque International Francophone sur l'Ecrit et le Document, Hammamet, Tunisia, 2002.

Evaluation: Procedure & Protocol

- Consider only Qery-by-String (QbS) scenario here
- Follow "Almazan" protocol (Almazan *et al.* 2014)
  - Every *unique transcription* in the test set is used as query
  - For experiments on IAM-DB: discard stop words
- Query PHOC $\boldsymbol{a}^q$ can be given directly
- PHOCNet predicts PHOC $\hat{\boldsymbol{a}}$ for each word image in the test set
- Test word images are ranked according to cosine dissimilarity to query

$$d_{\cos}(\hat{\boldsymbol{a}}, \boldsymbol{a}^q) = 1 - \frac{\hat{\boldsymbol{a}}^T \boldsymbol{a}^q}{||\hat{\boldsymbol{a}}|| \cdot ||\boldsymbol{a}^q||}$$

Evaluation Results

Segmentation-based Word Spotting Performance in mAP [%]

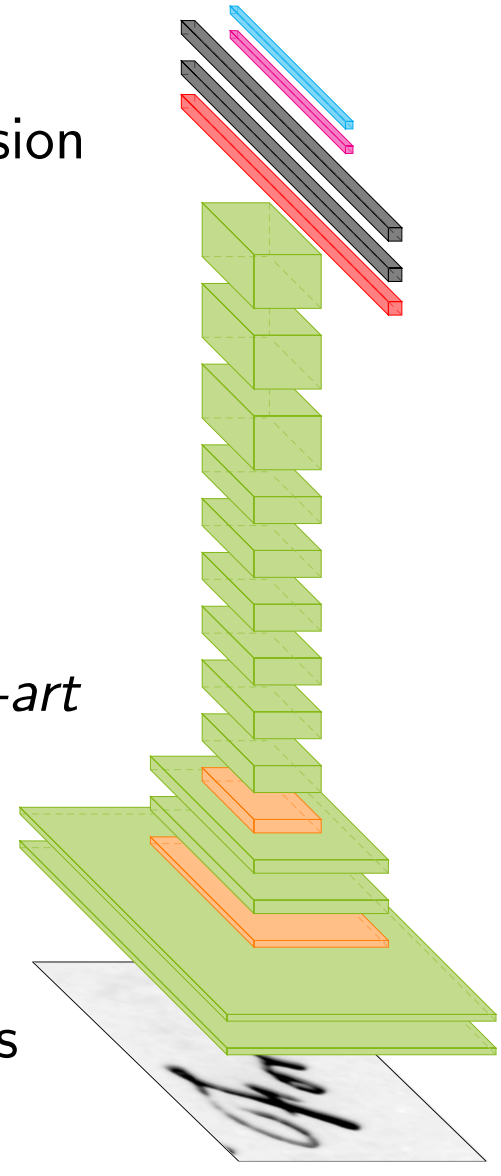
Method	GW	IAM	Esposalles	IFN/ENIT
TPP-PHOCNet	**97.92**	**93.42**	94.32	94.53
PHOCNet	97.44	**91.12**	**94.89**	93.87
Deep Feature Embedding [13]	92.84	91.58	–	–
Attribute SVM + FV [4]	91.29	73.72	–	–
LSA Embedding [1]	56.54	–	–	–
Triplet-CNN (*) [38]	93.69	89.49	–	–
BLSTM (*) [5]	84.00	78.00	–	–

Sudholt, S., Fink, G. A.: Attribute CNNs for Word Spotting in Handwritten Documents, Int. Journal on Document Analysis and Recognition, 2018, to appear.

Overview

Summary

- Deep Learning is a hot topic in Computer Vision *and* Document Image Analsis
  - ⇒ *You should know about it!*
- CNNs are the most popular deep networks
  - ⇒ *Especially suitable for (document) images!*
- Deep Networks are currently the *state-of-the-art* methods for word spotting
  - ⇒ *Especially the PHOCNet :-)*
- Toolboxes (e.g. Pytorch, Caffe) make it easy to apply Deep Learning to ones own problems
  - ⇒ *Beware of blindly using them!*



Outlook

Further Reading:

- Sudholt, S., Fink, G. A.: Attribute CNNs for Word Spotting in Handwritten Documents, Int. Journal on Document Analysis and Recognition, Special Issue on Deep Learning for Document Analysis and Recognition, 2018, to appear.

More about the PHOCNet:

- Monday, Oral Session 2 *Word/Object Spotting* (2 PM)
  Eugen Rusakov, L. Rothacker, H. Mo and G. A. Fink: *A Probabilistic Retrieval Model for Word Spotting Based on Direct Attribute Prediction*

References I

[1] David Aldavert, Marcal Rusinol, Ricardo Toledo, and Josep Llados.
Integrating visual and textual cues for query-by-string word spotting.
In *International Conference on Document Analysis and Recognition*, pages 511–515, 2013.

[2] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny.
Segmentation-free word spotting with exemplar svms.
Pattern Recognition, 47(12):3967 – 3978, 2014.

[3] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny.
Word spotting and recognition with embedded attributes.
IEEE Trans. on Pattern Analysis and Machine Intelligence, 36(12):2552–2566, 2014.

References II

[4] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny.
Word Spotting and Recognition with Embedded Attributes.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(12):2552–2566, 2014.

[5] Volkmar Frinken, Andreas Fischer, R. Manmatha, and Horst Bunke.
A novel word spotting method based on recurrent neural networks.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 34:211–224, 2012.

References III

[6] Xavier Glorot and Yoshua Bengio.
Understanding the difficulty of training deep feedforward neural networks.
AISTATS, 9:249–256, 2010.

[7] Phillip Good.
Permutation Tests - A Practical Guide to Resampling Methods for Testing Hypothesis.
Springer, 2 edition, 2000.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
Deep Learning.
The MIT Press, 2016.

References IV

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.
Spatial pyramid pooling in deep convolutional networks for visual recognition.
European Conference on Computer Vision, pages 346–361, 2014.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.
Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.
In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, pages 1026–1034, Washington, DC, USA, 2015.
initialization of deep neural networks termed 'MSRA' (Microsoft Research Asia) in Caffe.

References V

[11] Kurt Hornik, Maxwell Stinchcombe, and Halbert White.
Multilayer feedforward networks are universal approximators.
Neural Networks, 2(5):359–366, 1989.

[12] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell.
Caffe: Convolutional architecture for fast feature embedding.
arXiv preprint arXiv:1408.5093, 2014.

[13] Praveen Krishnan, Kartik Dutta, and C.V. Jawahar.
Deep feature embedding for accurate recognition and retrieval of handwritten text.
In *International Conference on Frontiers in Handwriting Recognition*, pages 289–294, 2016.

References VI

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton.
Imagenet classification with deep convolutional neural networks.
In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Neural Information Processing Systems*, pages 1097–1105, 2012.

[15] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce.
Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories.
In *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, 2006.

References VII

[16] Yann LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel.
Handwritten digit recognition with a back-propagation network.
Neural Information Processing Systems, pages 396–404, 1990.

[17] Josep Lladós, Marçal Rusiñol, Alicia Fornés, David Fernández, and Anjan Dutta.
On the influence of word representations for handwritten word spotting in historical documents.
Int. J. Pattern Recognition and Artificial Intelligence, 26(5), 2012.

[18] David Lowe.
Distinctive image features from scale-invariant keypoints.
Int. J. of Computer Vision, 60(2):91–110, 2004.

References VIII

[19] R. Manmatha, Chengfeng Han, E. M. Riseman, and W. B. Croft.
Indexing handwriting using word matching.
In *Proc. of the First ACM Int. Conf. on Digital Libraries*, DL '96, pages 151–159, New York, NY, USA, 1996. ACM.

[20] U.-V. Marti and H. Bunke.
The IAM-database: An English sentence database for offline handwriting recognition.
Int. Journal on Document Analysis and Recognition, 5:39–46, 2002.

[21] Stephen O'Hara and Bruce A. Draper.
Introduction to the bag of features paradigm for image classification and retrieval.
Computing Research Repository, arXiv:1101.3354v1, 2011.

References IX

[22] M. Pechwitz, S. Snoussi Maddouri, V. Märgner, N. Ellouze, and H. Amiri.
IFN/ENIT-database of handwritten Arabic words.
In *Proc. 7th Colloque International Francophone sur l'Ecrit et le Document*, Hammamet, Tunisia, October 2002.

[23] Arik Poznanski and Lior Wolf.
Cnn-n-gram for handwriting word recognition.
In *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, pages 2305–2314, Las Vegas, USA, 2016.

References X

[24] Toni M. Rath and R. Manmatha.
Word image matching using dynamic time warping.
In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–521–II–527 vol.2, June 2003.

[25] Verónica Romero, Alicia Fornès, Nicolás Serrano, Joan Andreu Sánchez, Alejandro H. Toselli, Volkmar Frinken, Enrique Vidal, and Josep Llad'os.
The esposalles database: An ancient marriage license corpus for off-line handwriting recognition.
Pattern Recognition, 46(6):1658 – 1669, 2013.

References XI

[26] F. Rosenblatt.
The perceptron: A probabilistic model for information storage and organization in the brain.
Psychological Review, 65(6):386–408, 1958.

[27] Leonard Rothacker and Gernot A. Fink.
Segmentation-free query-by-string word spotting with bag-of-features HMMs.
In *Proc. Int. Conf. on Document Analysis and Recognition*, Nancy, France, 2015.

References XII

[28] Leonard Rothacker, Denis Fisseler, Gerfrid G.W. Müller, Frank Weichert, and Gernot A. Fink.
Retrieving cuneiform structures in a segmentation-free word spotting framework.
In *Proc. of the Int. Workshop on Historical Document Imaging and Processing*, Nancy, France, 2015.

[29] Leonard Rothacker, Marcal Rusinol, and Gernot A. Fink.
Bag-of-features HMMs for segmentation-free word spotting in handwritten documents.
In *Proc. Int. Conf. on Document Analysis and Recognition*, Washington DC, USA, 2013.

References XIII

[30] Leonard Rothacker, Marcal Rusinol, Josep Llados, and Gernot A. Fink.
A Two-Stage Approach to Segmentation-Free Query-by-Example Word Spotting.
manuscript cultures, 1(7):47–57, 2014.

[31] Leonard Rothacker, Szilard Vajda, and Gernot A. Fink.
Bag-of-features representations for offline handwriting recognition applied to Arabic script.
In *Proc. Int. Conf. on Frontiers in Handwriting Recognition*, Bari, Italy, 2012.

References XIV

[32] M. Rusiñol, D. Aldavert, R. Toledo, and J. Llados.
Browsing heterogeneous document collections by a segmentation-free word spotting method.
In *Proc. Int. Conf. on Document Analysis and Recognition*, pages 63 –67, Bejing, China, 2011.

[33] R. Shekhar and C.V. Jawahar.
Word image retrieval using bag of visual words.
In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, pages 297–301, March 2012.

[34] Karen Simonyan and Andrew Zisserman.
Very deep convolutional networks for large-scale image recognition.
arXiv, pages 1–13, 2014.

References XV

[35] Sebastian Sudholt and Gernot A. Fink.
PHOCNet: A deep convolutional neural network for word spotting in handwritten documents.
In *Proc. Int. Conf. on Frontiers in Handwriting Recognition*, Shenzhen, China, 2016.
Winner of the best paper award.

[36] Sebastian Sudholt and Gernot A. Fink.
Evaluating word string embeddings and loss functions for cnn-based word spotting.
In *Proc. Int. Conf. on Document Analysis and Recognition*, Kyoto, Japan, 2017.

References XVI

[37] Sebastian Sudholt and Gernot A. Fink.
Attribute cnns for word spotting in handwritten documents.
Int. Journal on Document Analysis and Recognition, 2018.
to appear.

[38] Tomas Wilkinson and Anders Brun.
Semantic and verbatim word spotting using deep neural networks.
In *International Conference on Frontiers in Handwriting Recognition*, pages 307–312, 2016.