Sequence Learning

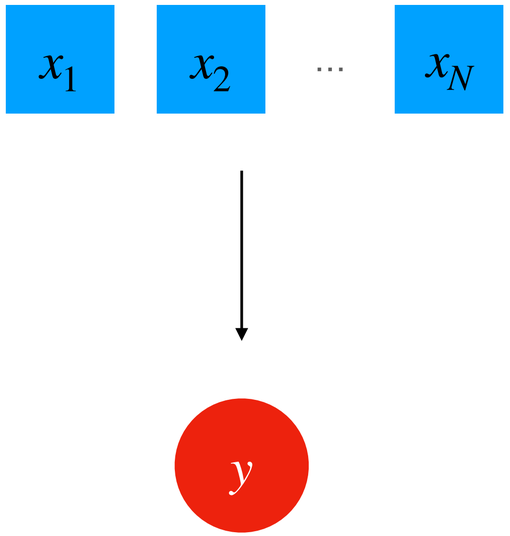
Feed-Forward Networks for Sequence Data

Korbinian Riedhammer

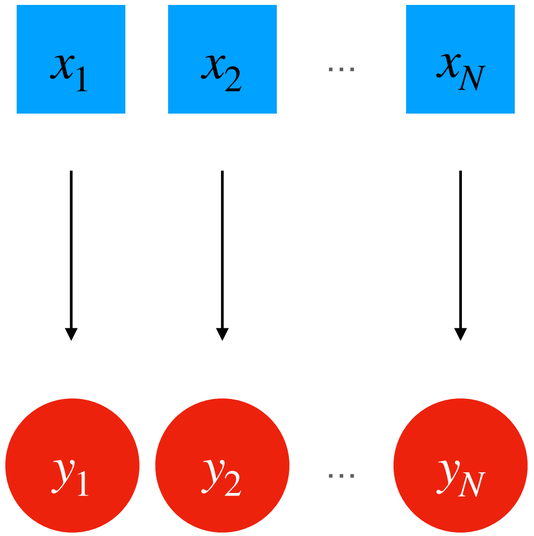
TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Feed-Forward Networks

…on sequence data



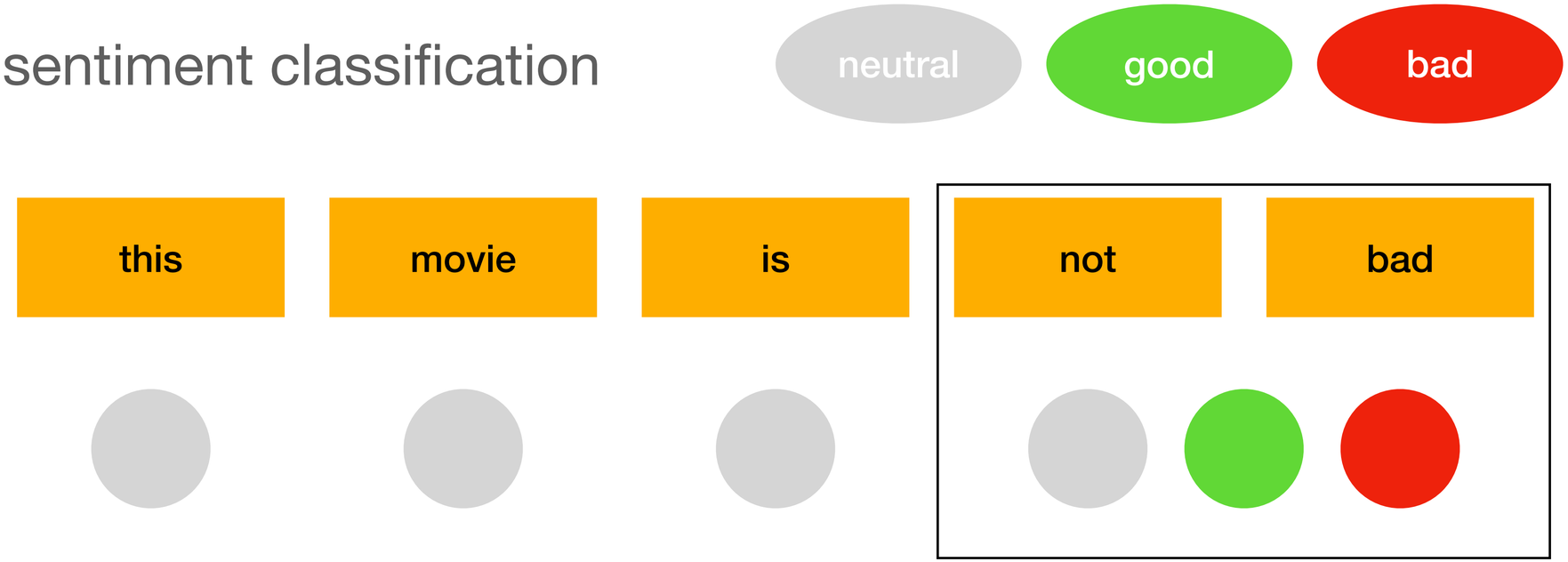
many-to-one



many-to-many

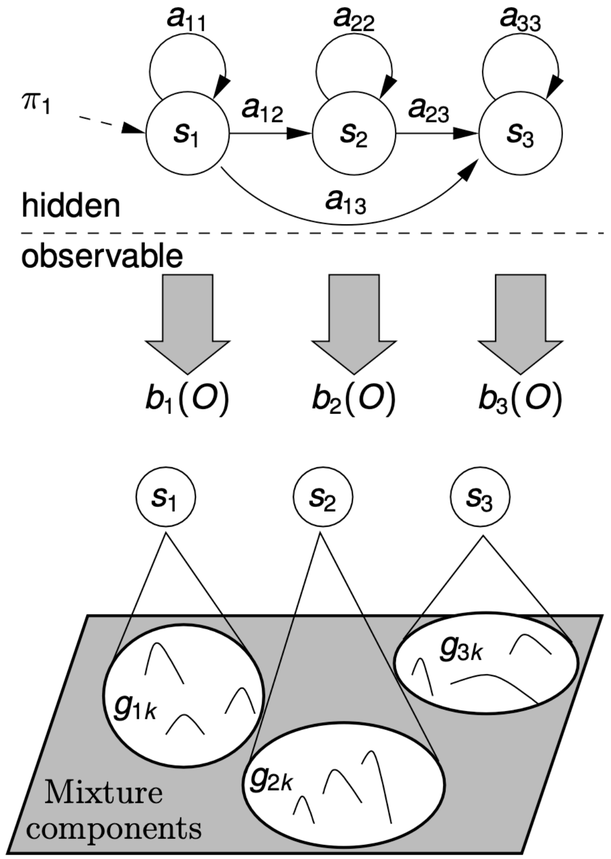
Context is Crucial

Example: sentiment classification



Solution: Use context windows to learn temporal relations

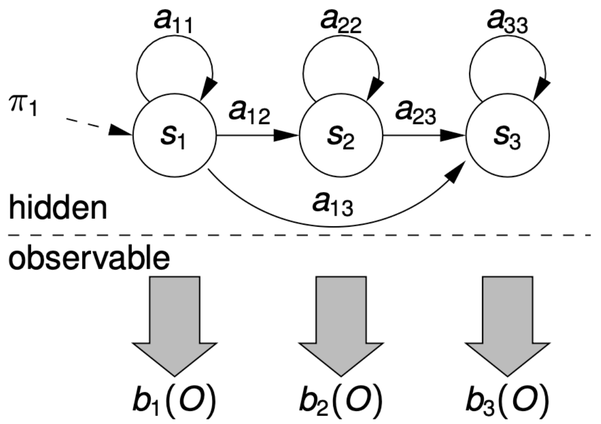
Connectionist HMM



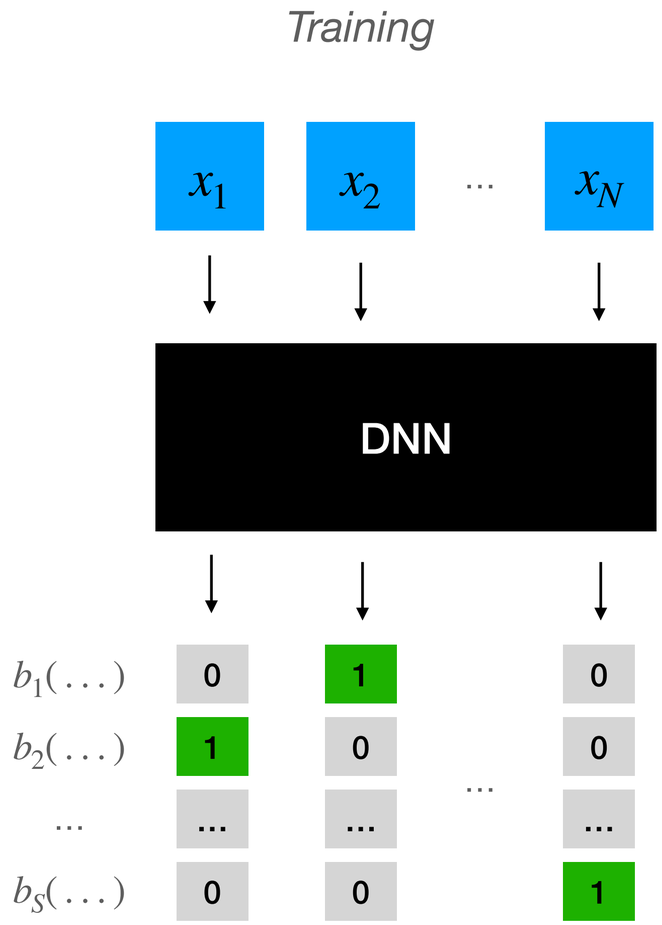
- *Observation*: At decoding time, we need emission probabilities of all (active) states
- *Problem*: GMMs don't generalize well
- *Idea*: Use NN to "predict" emission probs for all states at the same time

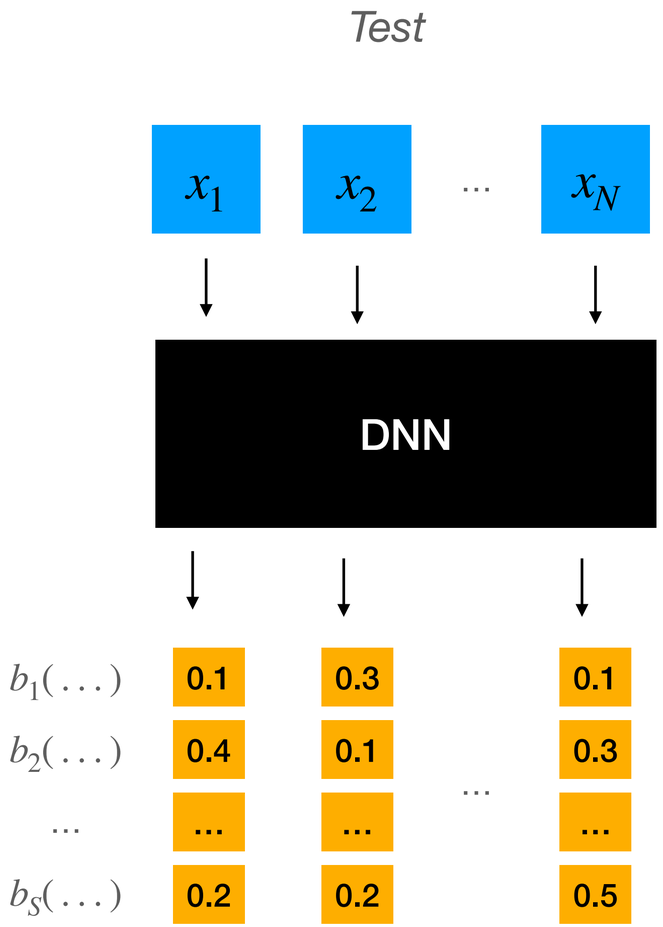
Renals et al., 1992: Connectionist Probability Estimation in the DECIPHER Speech Recognition System

Connectionist HMM



- requires alignment
- “one-hot encoding”
- cross-entropy loss



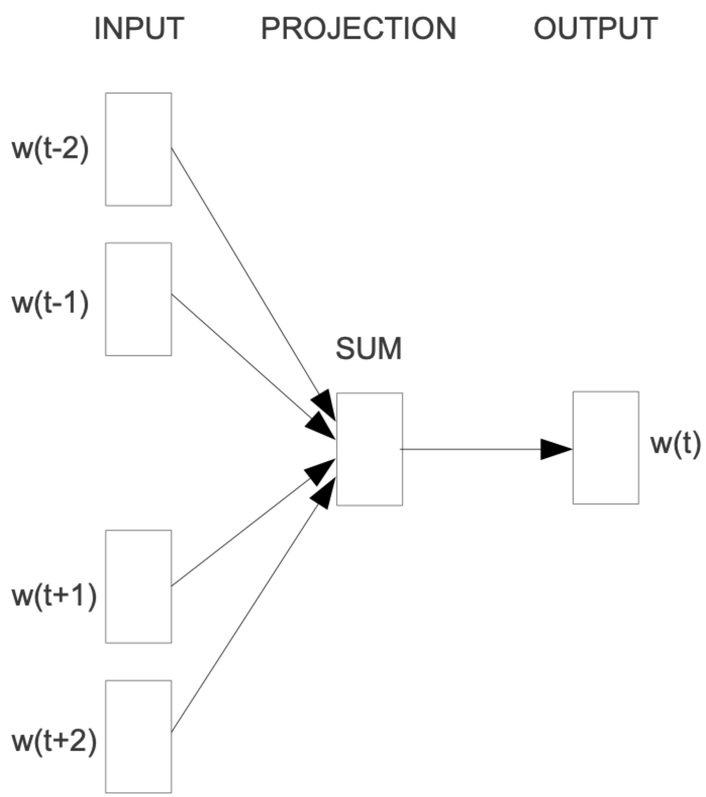


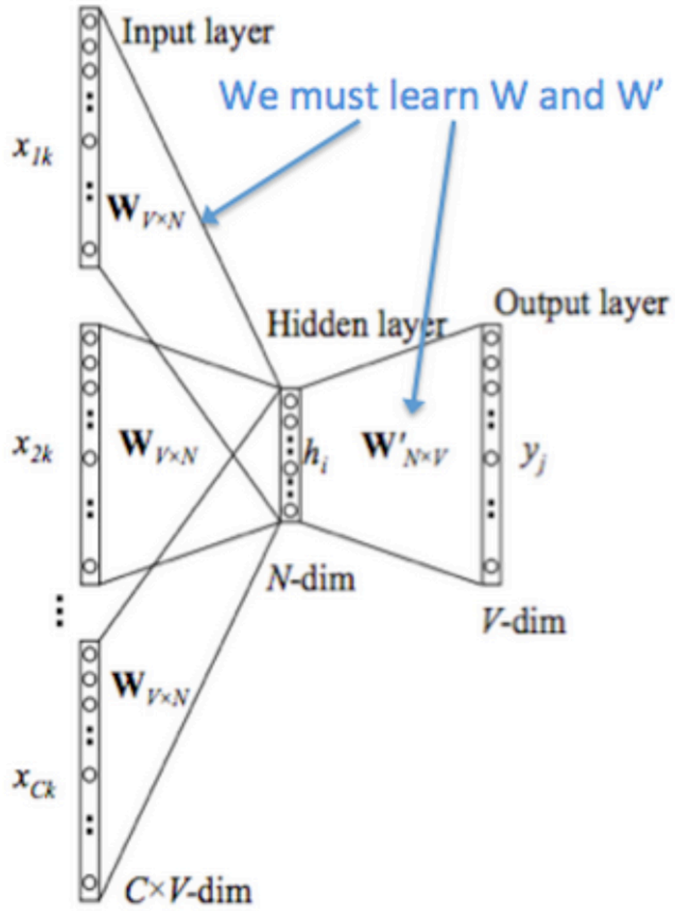
Word2Vec

- Recall n-gram probabilities: count observed ngrams, use back-off for unseen
- Bi-grams probabilities limit the context: $P(w_1, w_2, \ldots, w_n) = P(w_1) \prod_{i=2}^{N} P(w_i \mid w_{i-1})$
- How could we learn (not count) these?

Continuous Bag of Words (CBOW)

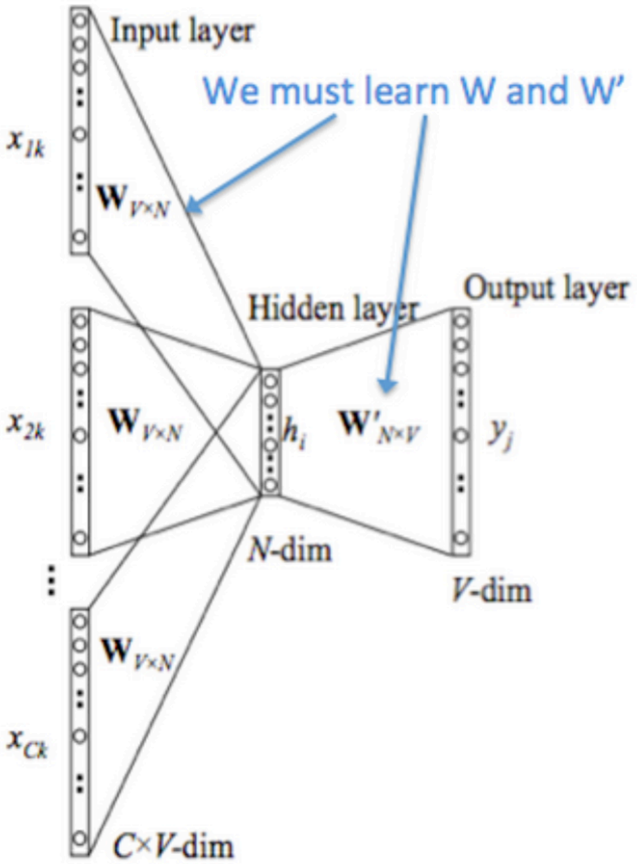
Predict center word given context





Mikolov et al., 2013. “Efficient Estimation of Word Representations in Vector Space”

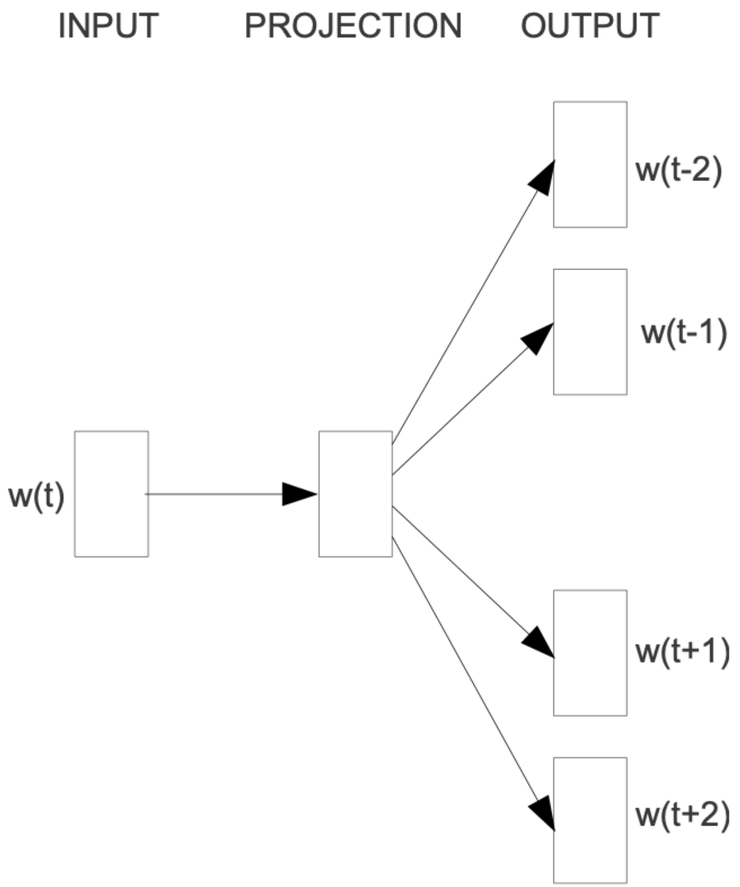
CBOW cont'd

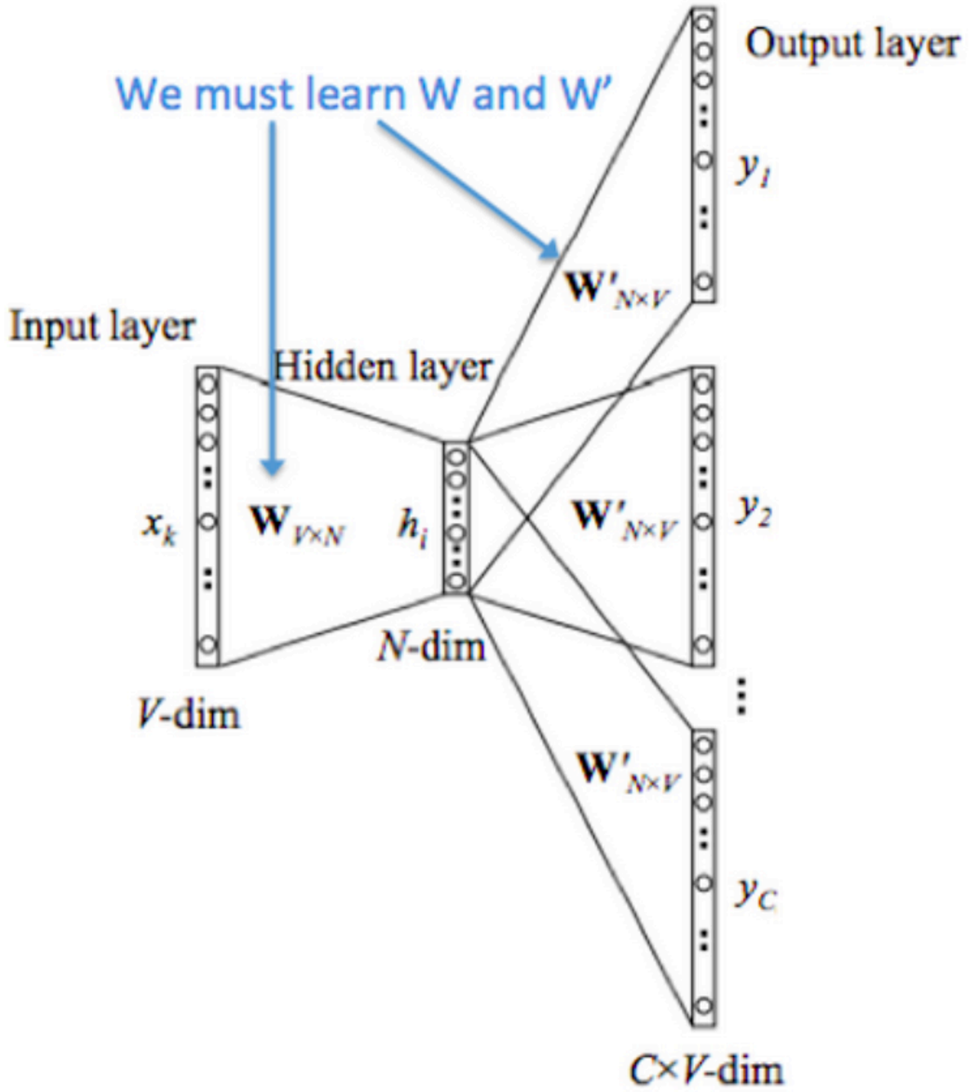


1. Generate 1-hot vectors for context
2. Compute embedded word vectors $v_i = Vx_i$
3. Compute average $\hat{v} = \frac{1}{2m}\sum v_i$
4. Generate score vector $z = U\hat{v}$
5. Turn scores into probabilities $\hat{y} = \text{softmax}(z)$; 1-hot in training

Skip-gram

Given a word, predict its context





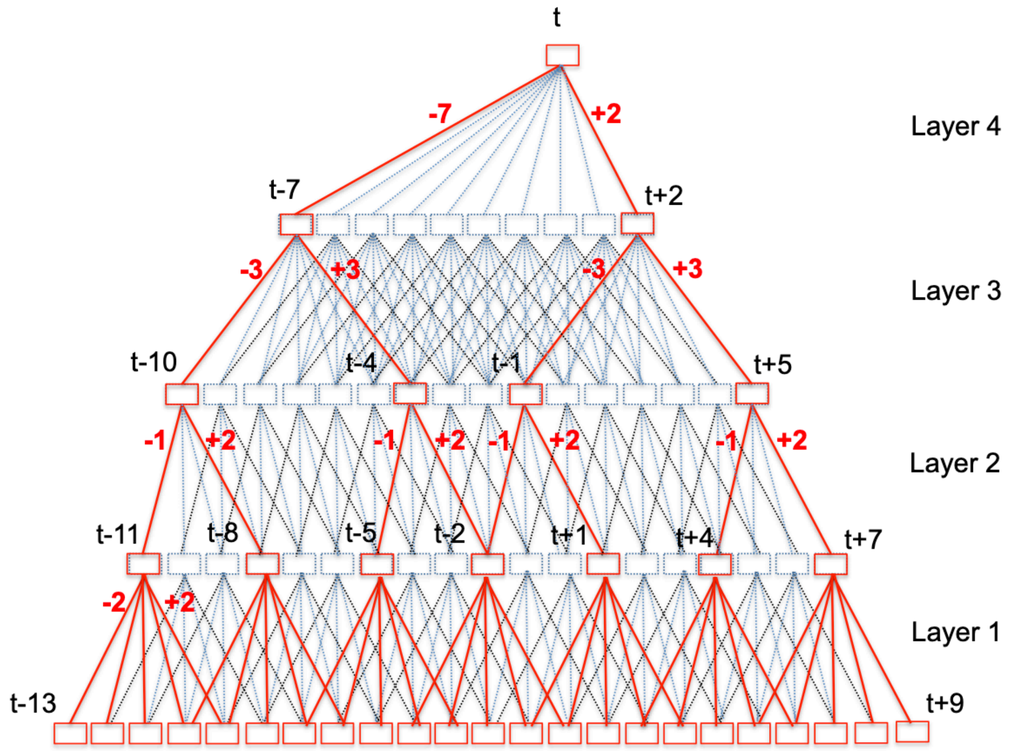
CBOW, Skip-Gram, Generalization

Remarks

- Essentially, word2vec learns two matrices (and thus two vectors per word)
  - Use either as embedding,
  - …or their mean value,
  - …or concatenate them.
- Generalization:
  - Use *embedding layer* to encode discrete symbols (eg. using nn.Embedding)
  - Learn embedding as part of regular training

Time-delay Neural Networks

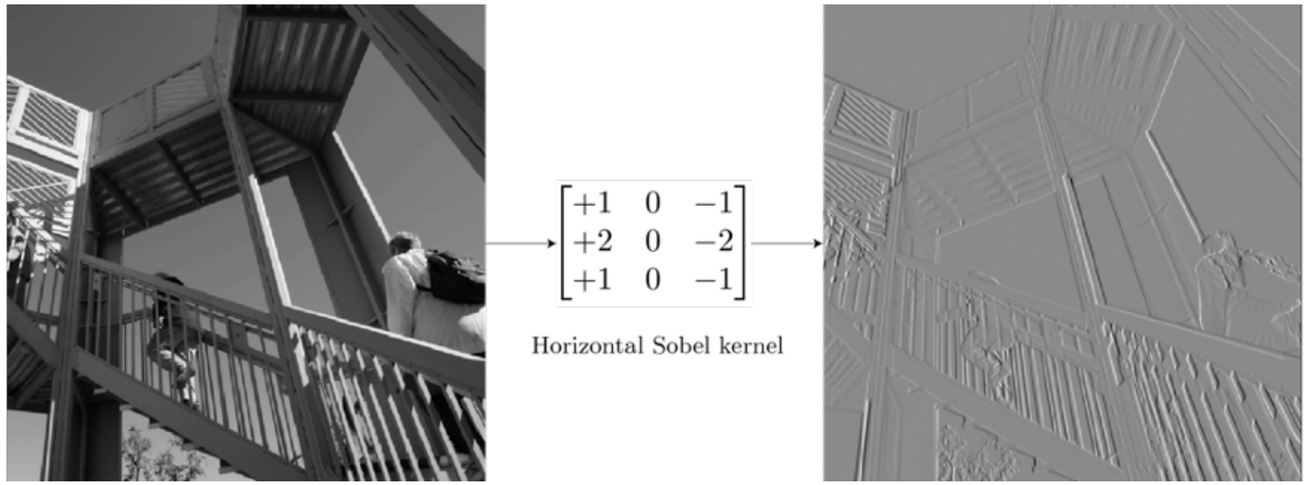
Waibel et al. 1989



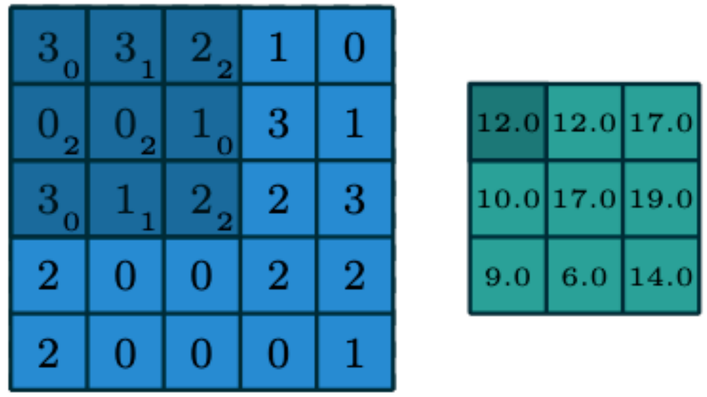
- Frames are typically features (MFCC, word embeddings, …)
- Concatenate frames to form contexts
- Go from narrow to wide with layers
- Lower layers learn “local” features
- Higher layers learn temporal relationships

Peddinti et al., 2015. “A time delay neural network architecture for efficient modeling of long temporal contexts”

ConvNets



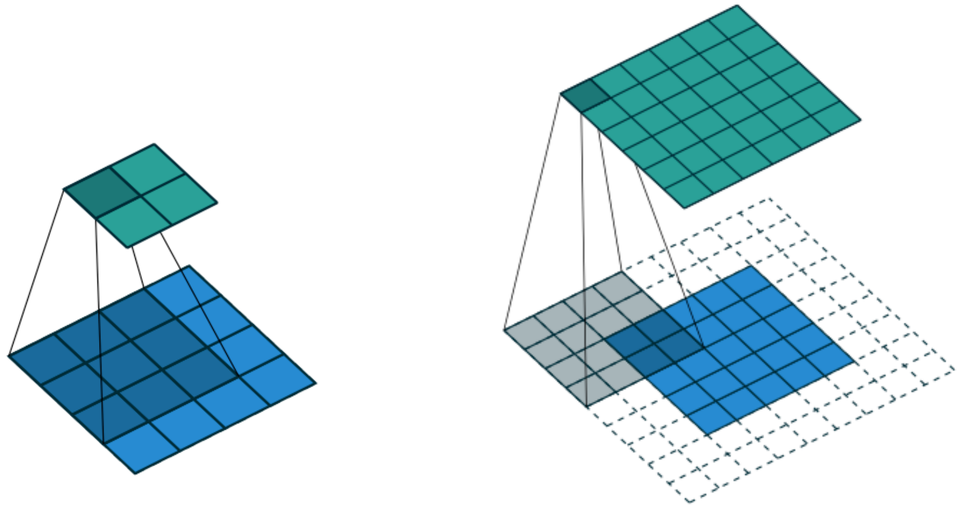
- Motivation:
  - Convolution of signal with special kernels can be a great feature
  - Well established in computer graphics (eg. Sobel edge detector)
- 1D time series: 1D convolutions
  - “within-feature convolutions”
- 2D image: 2D convolutions
  - “across-feature convolutions”

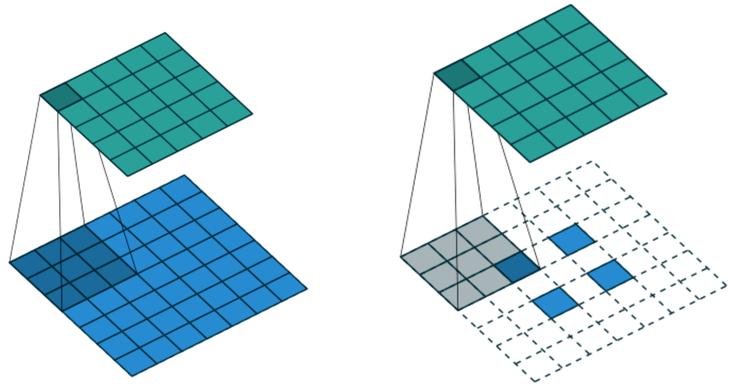


Dumoulin, V. and Visin, F. “A guide to convolution arithmetic for deep learning”

ConvNets Building Blocks

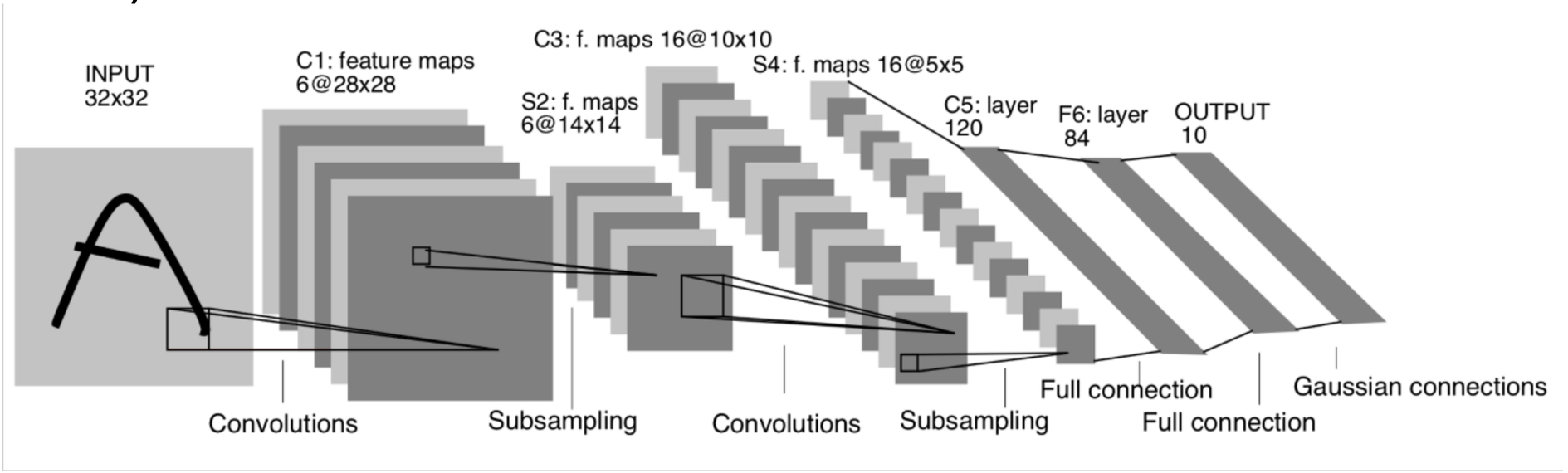
- Convolution:
  - kernel size, eg. 3x3, 1x3
  - stride, step size, eg. 1
  - padding, what to do at the edges? eg. zero-pad
- Pooling to reduce/increase resolution
  - average, max, …





Historic Note

- TDNN (1989): effectively 1D convolutions
- LeCun at al., 1998: LeNet-5 architecture, MNIST error rate 0.8% (regular FF: 4.7%)



Recap

Feed-Forward Networks for Sequence Data

- Use context windows, eg. by concatenation
- Use embeddings for discrete symbols (which effectively use 1-hot)
- Use convolutions (1D, 2D) to extract temporal structure from context window
- Works for all modalities:
  - Audio: eg. MFB, MFCC
  - Text: Word Vectors