Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Master of Science (MSc)

Applied Information and Data Science

Institut für Kommunikation und Marketing IKM
Dr. Manuel Dömer
Externer Dozent

T direkt +41 79 551 64 17
manuel.doemer@hslu.ch

Luzern 06.05.20

Computational Language Technologies

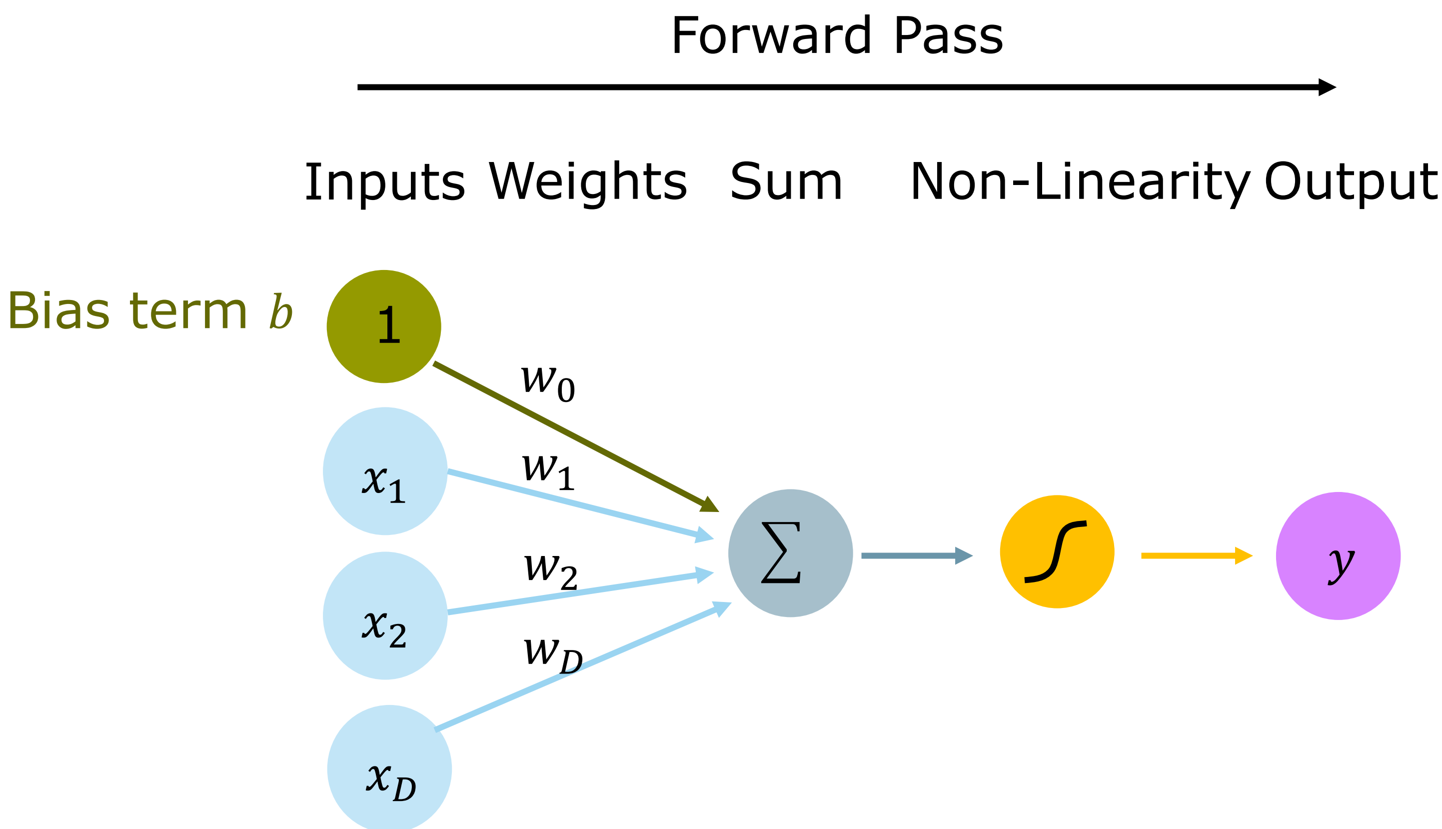
Neural Language Models

- Neural Language Models use neural networks to compute the probability distributions of sequences of words
- Learn distributed representations for words: word embeddings
- Word2vec
- Compared to n-gram models: no need for smoothing, larger context and better generalization.
  Comparably slow in training and prediction and need larger datasets
- Use for
  - Document Classification
  - Sequence Labelling
    - Language Translation
    - Part of Speech Tagging
    - Named Entity Recognition
    - Syntactic Parsing

Recap

FEED-FORWARD NEURAL NETWORKS

The Perceptron – Logistic Regression



$$z = w^T \cdot x + w_0 \qquad \sigma(z) = \frac{1}{1 + e^{-z}}$$

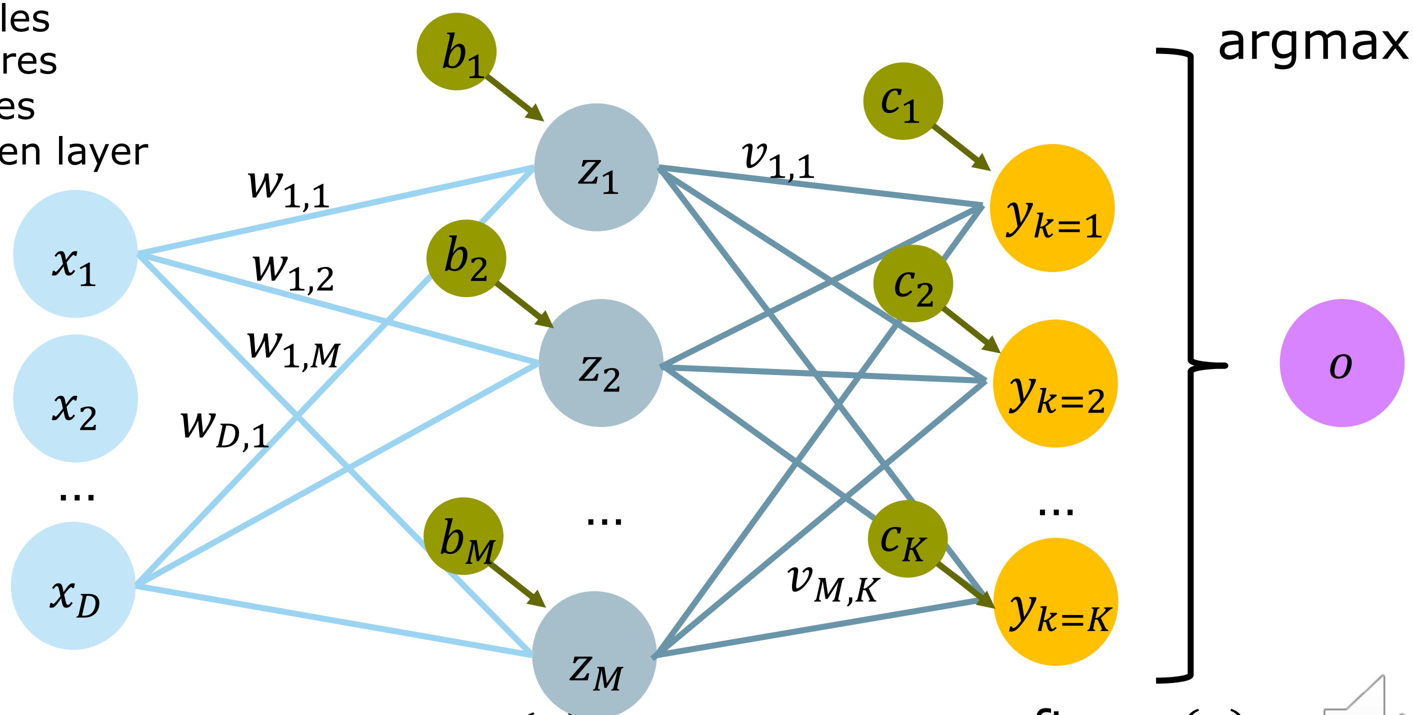
Feed-Forward Neural Network

Variables
(Dimensions)

Samples $X: N \times D$
N: Number of Samples
D: Number of Features
K: Number of Classes
M: units in the hidden layer

Forward Pass on sample x (1 row of X)

Inputs	Weights	Hidden Layer			Output Layer	
$x(D)$	$W(D \times M)$	$b(M)$	$z(M)$	$V(M \times K)$	$c(K)$	$y(K)$



$$z = \sigma(\alpha)$$
$$\alpha = W^T \cdot x + b$$

$$y_k = \text{softmax}(a)$$
$$a = V^T \cdot z + c$$

Parameter Optimisation

- Categorical Cross-Entropy Loss

$$L_{CE} = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_{nk}$$

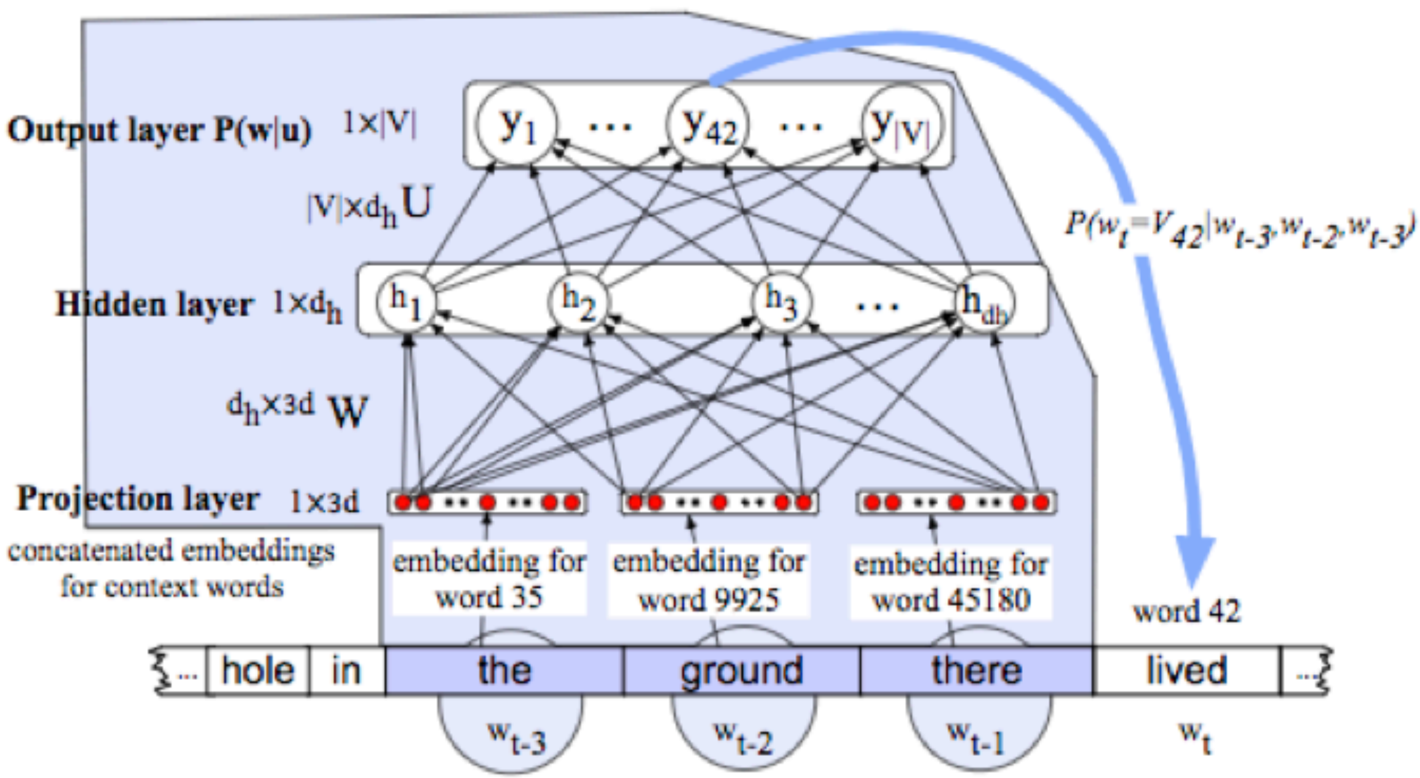
- Back-Propagation
- Gradient Descent

WORD2VEC

Word2Vec

- Mikolov and friends in 2013: Efficient estimation of word representations in vectorspace InICLR 2013 and Distributed representations of words and phrases and their compositionality in Advances in Neural Information Processing Systems
- 2 variants: skip-gram and continuous bag of words (CBOW)
- train a shallow feed-forward neural network to predict a word given a sequence of context words
- Projection Layer -> Word Embeddings

Word2Vec – NN Architecture



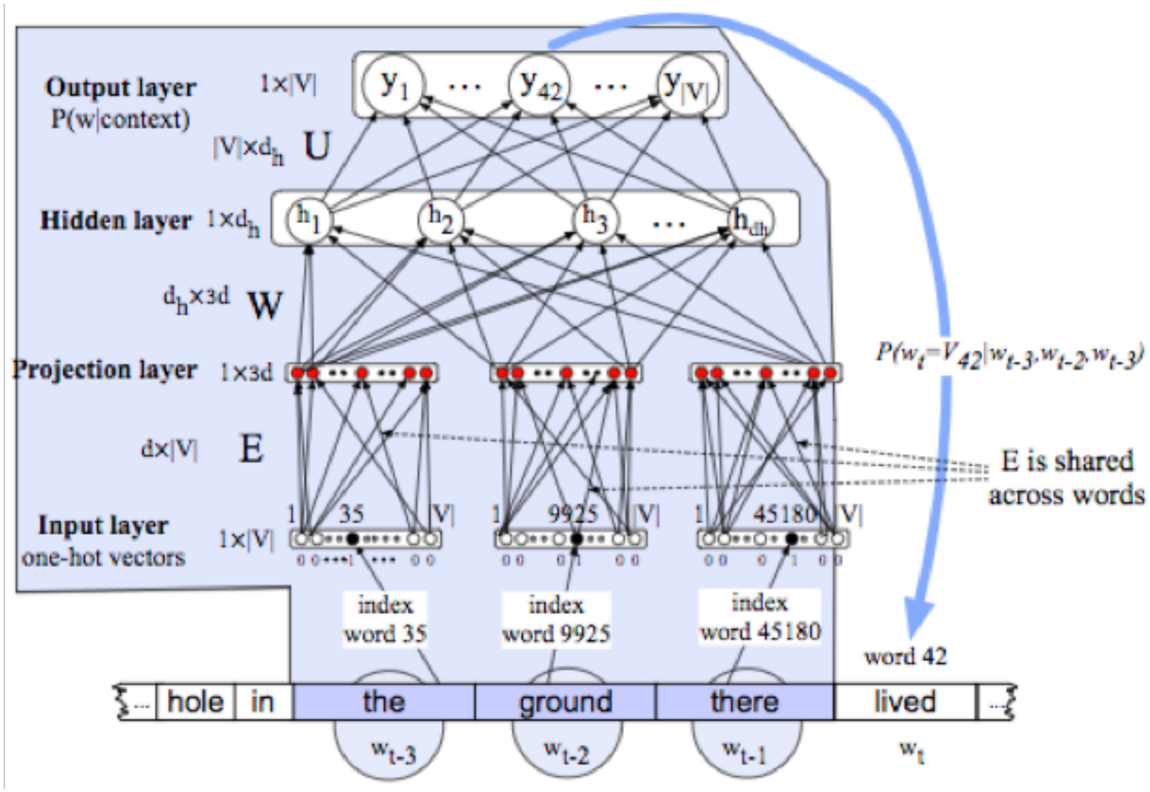
Learning the Word Embeddings

$$y = \mathrm{softmax}(z)$$

$$z = Uh$$

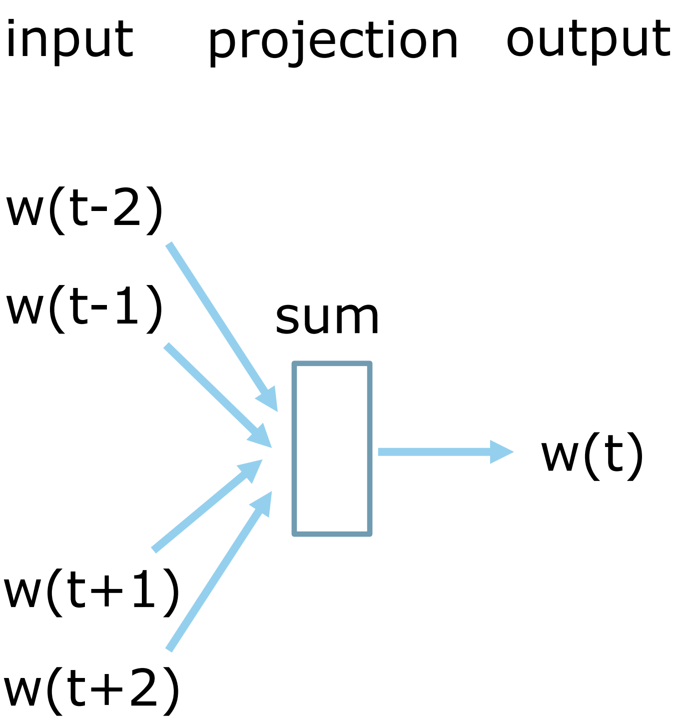
$$h = \sigma(We + b)$$

$$e = (Ex_1, Ex_2, \ldots, Ex_V)$$

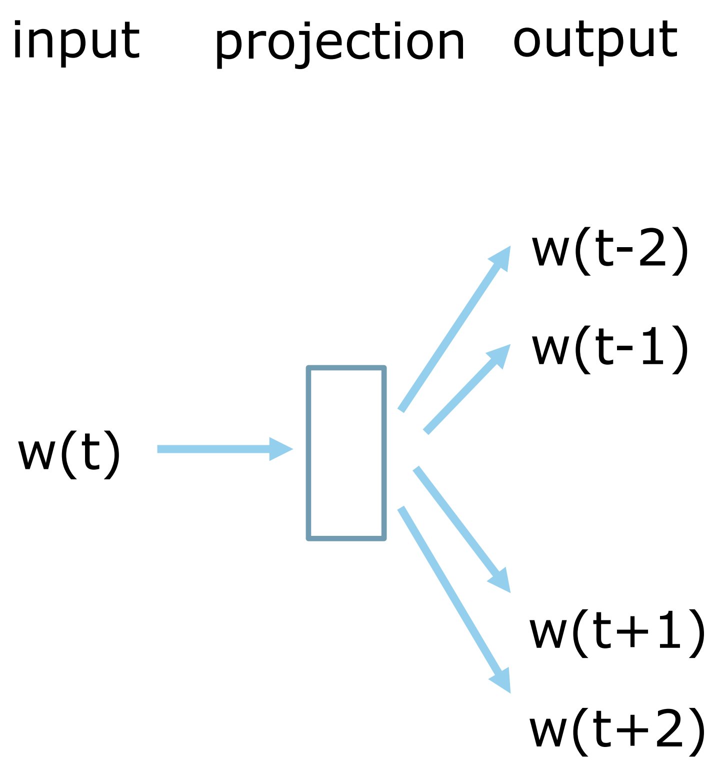


The Classification Task

CBOW



Skip-Gram



Skip-Gram

sliding context window: target context

… hole in the ground there lived… → (the, hole) (the, in) (the, ground) (the, there)

… hole in the ground there lived … → (ground, in) (ground, the) (ground, there) (ground, lived)

training samples (word pairs):

(the, hole)
(the, in)
(the, ground)
(the, there)

(ground, in)
(ground, the)
(ground, there)
(ground, lived)

The Classification Task

CBOW

- predicts target word from context window
- for each occurrence of the target word treats the context as one observation
  -> low probability for infrequent target words
- several times faster to train than
- slightly better accuracy for the frequent words.

Skip-Gram

- Given the «target word» predicts for each word in the vocabulary the probability that it can be found in the context window
- works well with a small amount of the training data
- represents well even rare words or phrases

Properties of the Word Embeddings

- dense representation of the words, in contrast to one-hot encoding
  Neural Networks do not perform well on sparse vectors
- the NN is trained to predict similar contexts for words with similar context in the training data
- once trained, the corresponding word vectors of words with similar context are close in embedding space (and vice versa)
- represent more informative features than simple bag-of-words vectors