Machine Learning – Lecture 18

Word Embeddings

08.01.2020

Bastian Leibe

RWTH Aachen

http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Important Announcement

Happy New Year everybody!

B. Leibe

2

Course Outline

- ## Fundamentals
  - ➢ Bayes Decision Theory
  - ➢ Probability Density Estimation

- ## Classification Approaches
  - ➢ Linear Discriminants
  - ➢ Support Vector Machines
  - ➢ Ensemble Methods & Boosting
  - ➢ Random Forests

- ## Deep Learning
  - ➢ Foundations
  - ➢ Convolutional Neural Networks
 - ➢ Recurrent Neural Networks
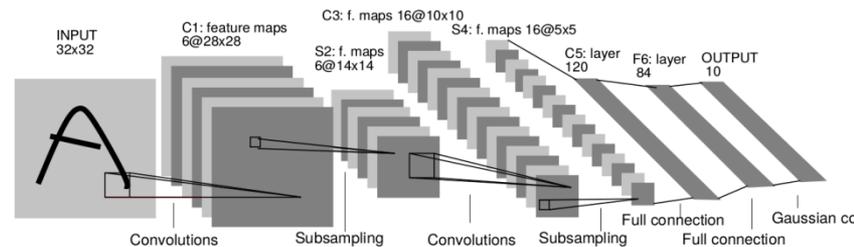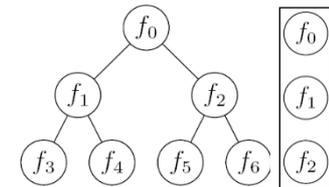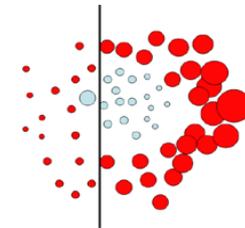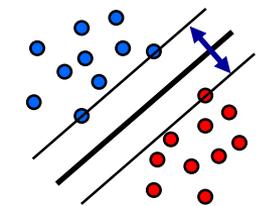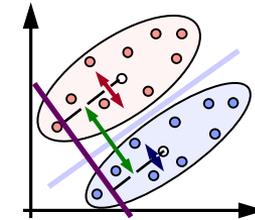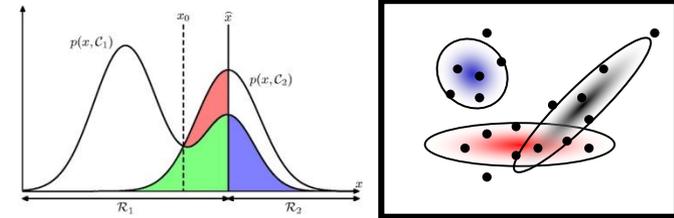
B. Leibe

Topics of This Lecture

- **Recap**
  - ResNets
  - Applications of CNNs

- Word Embeddings
  - Neuroprobabilistic Language Models
  - word2vec
  - GloVe
  - Hierarchical Softmax

- Embeddings in Vision
  - Siamese networks
 - Triplet loss networks

- Outlook: Recurrent Neural Networks

B. Leibe

Recap: Residual Networks

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

- Core component
  - Skip connections bypassing each layer
  - Better propagation of gradients to the deeper layers
  - This makes it possible to train (much) deeper networks.

x

weight layer

relu

$F(x)$

weight layer

$H(x) = F(x) + x$

relu

B. Leibe

6
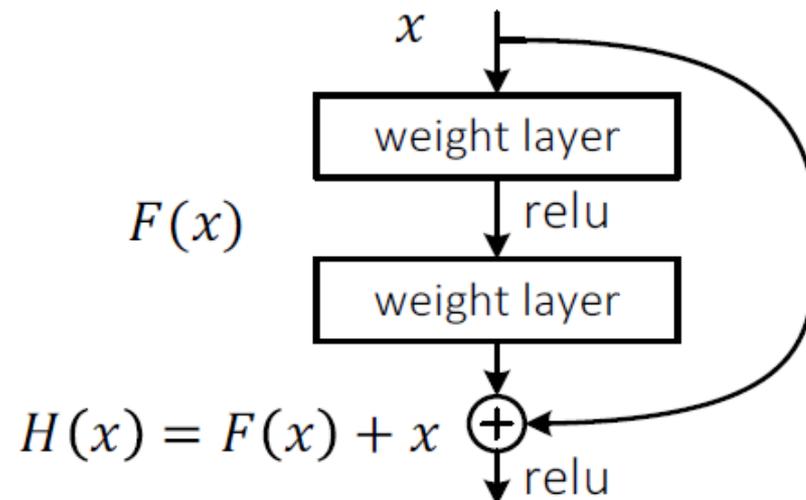
Recap: Analysis of ResNets

- ## The effective paths in ResNets are relatively shallow
  - ➤ Effectively only 5-17 active modules

- ## This explains the resilience to deletion
  - ➤ Deleting any single layer only affects a subset of paths (and the shorter ones less than the longer ones).

- ## New interpretation of ResNets
  - ➤ ResNets work by creating an ensemble of relatively shallow paths
  - ➤ Making ResNets deeper increases the size of this ensemble
  - ➤ Excluding longer paths from training does not negatively affect the results.

Image source: Veit et al., 2016

Machine Learning Winter '19

Slide credit: Ross Girshick

B. Leibe
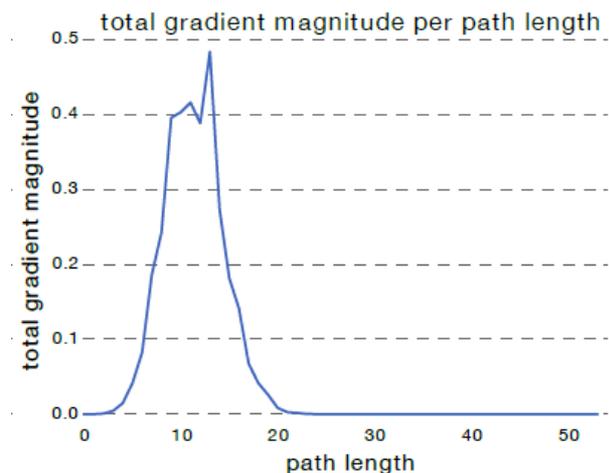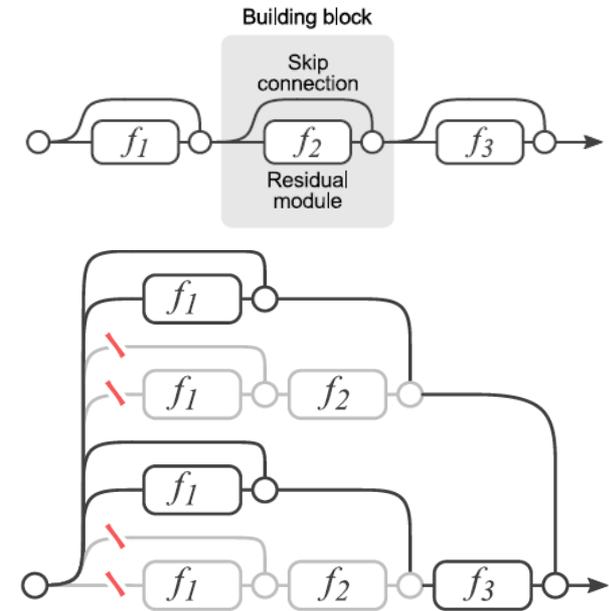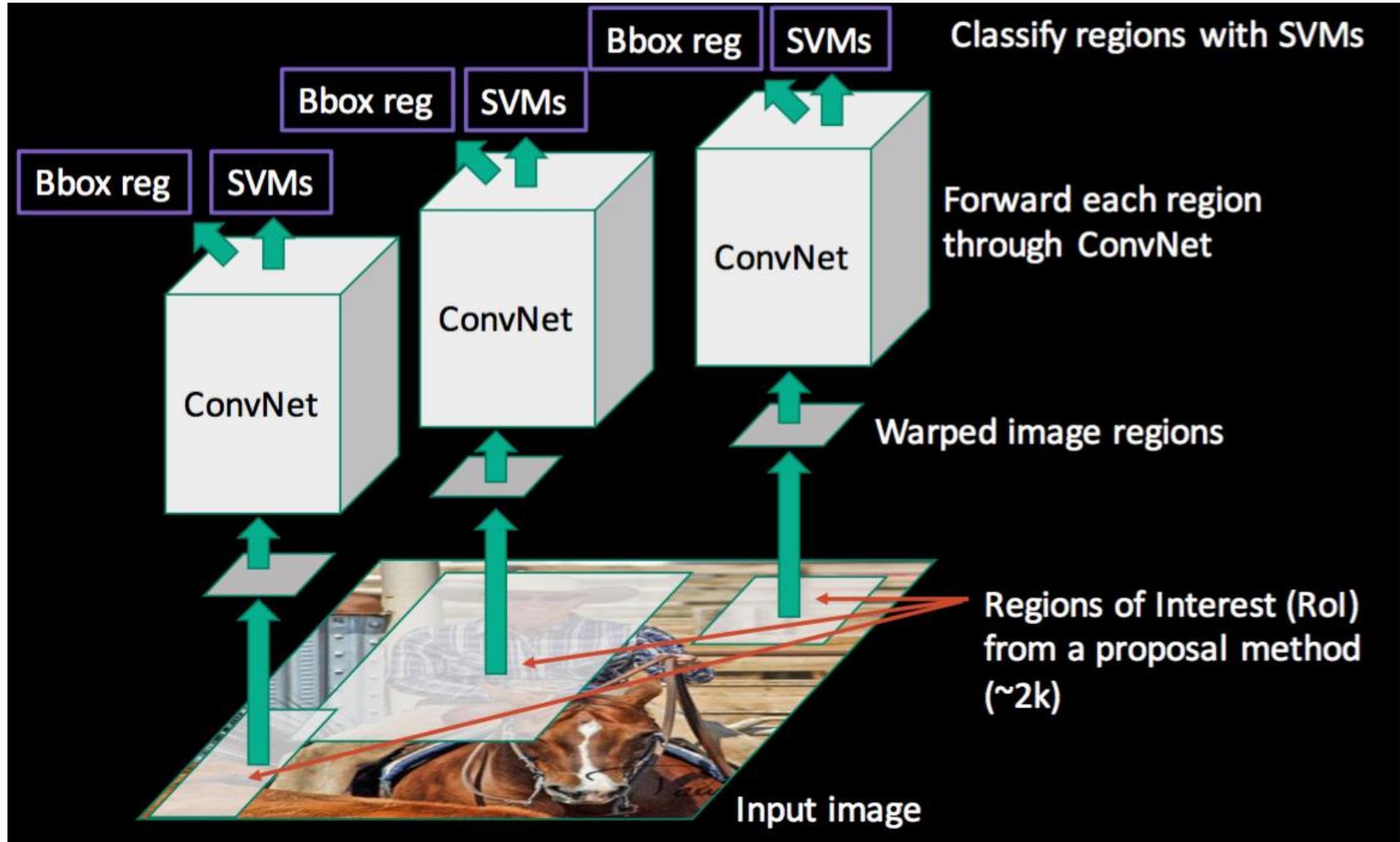
Recap: Faster R-CNN

- One network, four losses
  - ➢ Remove dependence on external region proposal algorithm.

  - ➢ Instead, infer region proposals from same CNN.
  - ➢ Feature sharing
  - ➢ Joint training
  - ⇒ Object detection in a single pass becomes possible.
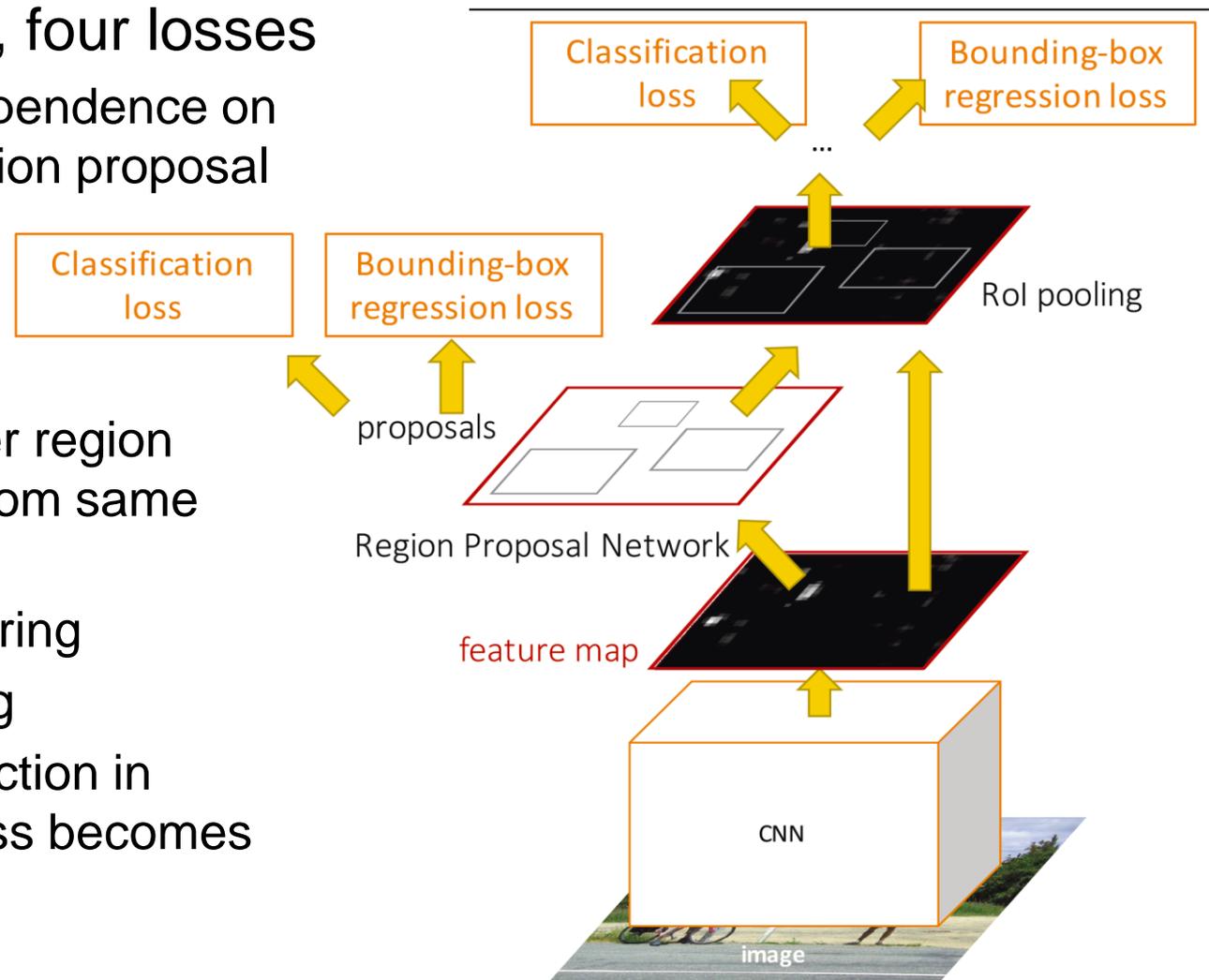
Slide credit: Ross Girshick

Topics of This Lecture

- **Recap**
  - ResNets
  - Applications of CNNs

- **Word Embeddings**
  - Neuroprobabilistic Language Models
  - word2vec
  - GloVe
  - Hierarchical Softmax

- **Embeddings in Vision**
  - Siamese networks
  - Triplet loss networks

- **Outlook: Recurrent Neural Networks**

Neural Networks for Sequence Data

- ## Up to now
  - ➢ Simple structure: Input vector $\rightarrow$ Processing $\rightarrow$ Output

- ## In the following, we will look at sequence data
  - ➢ Interesting new challenges
  - ➢ Varying input/output length, need to memorize state, long-term dependencies, ...

- ## Currently a hot topic
  - ➢ Early successes of NNs for text / language processing.
  - ➢ Very good results for part-of-speech tagging, automatic translation, sentiment analysis, etc.
  - ➢ Recently very interesting developments for video understanding, image+text modeling (e.g., creating image descriptions), and even single-image understanding (attention processes).

B. Leibe

Motivating Example

- **Predicting the next word in a sequence**
  - Important problem for speech recognition, text autocorrection, etc.

- **Possible solution: The trigram (n-gram) method**
  - Take huge amount of text and count the frequencies of all triplets (n-tuples) of words.
  - Use those frequencies to predict the relative probabilities of words given the two previous words

  $$\frac{p(w_3 = c | w_2 = b, w_1 = a)}{p(w_3 = d | w_2 = b, w_1 = a)} = \frac{\text{count}(abc)}{\text{count}(abd)}$$

  - State-of-the-art until not long ago...

B. Leibe

Machine Learning Winter '19

Problems with N-grams

- Problem: Scalability
    - We cannot easily scale this to large N.
    - The number of possible combinations increases exponentially
    - So does the required amount of data

- Problem: Partial Observability
    - With larger N, many counts would be zero.
    - The probability is not zero, just because the count is zero!
    $\Rightarrow$ Need to back off to (N-1)-grams when the count for N-grams is too small.
    $\Rightarrow$ Necessary to use elaborate techniques, such as Kneser-Ney smoothing, to compensate for uneven sampling frequencies.

B. Leibe

Let's Try Neural Networks for this Task

```
┌──────────────────────────────────────────────────────────────┐
│        "softmax" units (one per possible next word)            │
└──────────────────────────────────────────────────────────────┘
                               ↑
┌──────────────────────────────────────────────────────────────┐
│                    internal NN structure                       │
└──────────────────────────────────────────────────────────────┘
           ↑                                         ↑
┌──────────────────────────┐         ┌──────────────────────────┐
│   index of word at t-2   │         │   index of word at t-1   │
└──────────────────────────┘         └──────────────────────────┘
```
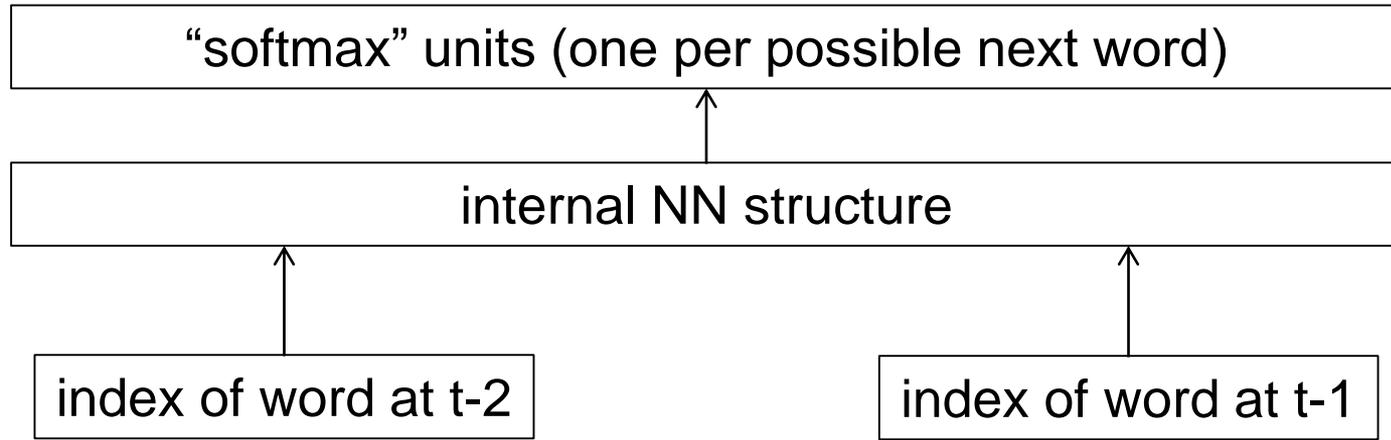
- Important issues
  - How should we encode the words to use them as input?
  - What internal NN structure do we need?
  - How can we perform classification (softmax) with so many possible outputs?

17

B. Leibe

Neural Probabilistic Language Model
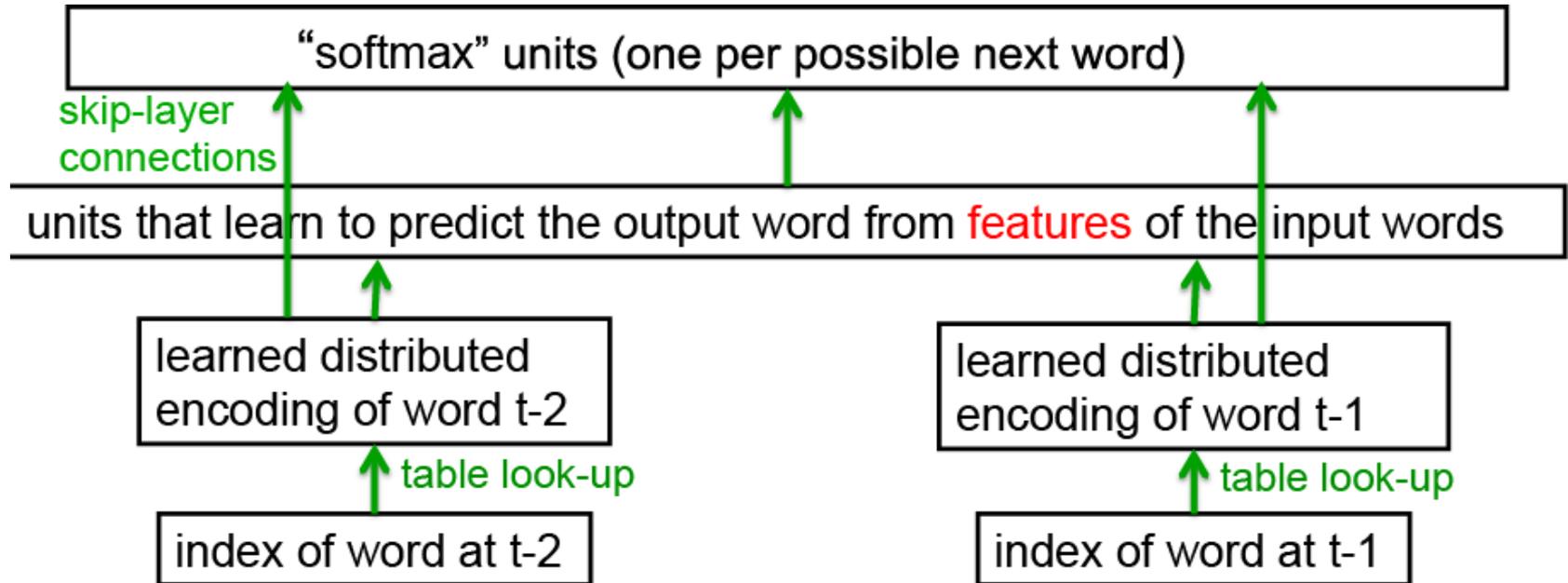
- **Core idea**
  - Learn a shared distributed encoding (word embedding) for the words in the vocabulary.

Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A Neural Probabilistic Language Model, In JMLR, Vol. 3, pp. 1137-1155, 2003.

B. Leibe

Word Embedding

- Idea
  - ➢ Encode each word as a vector in a d-dimensional feature space.
  - ➢ Typically, $V \sim 1\text{M}, d \in (50, 300)$

- Learning goal
  - ➢ Determine weight matrix $\mathbf{W}_{V \times d}$ that performs the embedding.
  - ➢ Shared between all input words

- Input
  - ➢ Vocabulary index $\mathbf{x}$ in 1-of-K encoding.
  - ➢ For each input $\mathbf{x}$, only one row of $\mathbf{W}_{V \times d}$ is needed.
  - $\Rightarrow \mathbf{W}_{V \times d}$ is effectively a look-up table.

B. Leibe

19

Image source: Xin Rong, 2015
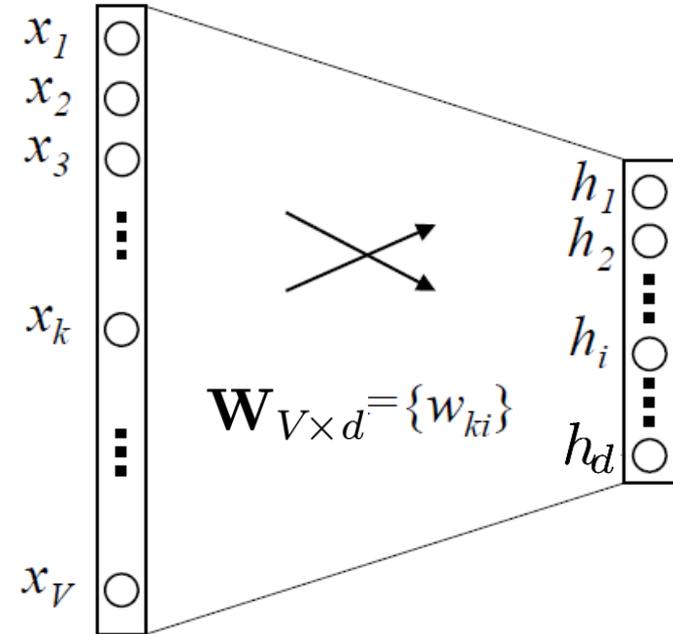
Word Embedding: Full Network

mapping to hidden units

skip connections

$\mathbf{W}_{2d \times V}$

$\mathbf{W}_{2d \times N}$

$\mathbf{W}_{V \times d} = \{w_{ki}\}$

$\mathbf{W}'_{N \times V} = \{w'_{ij}\}$
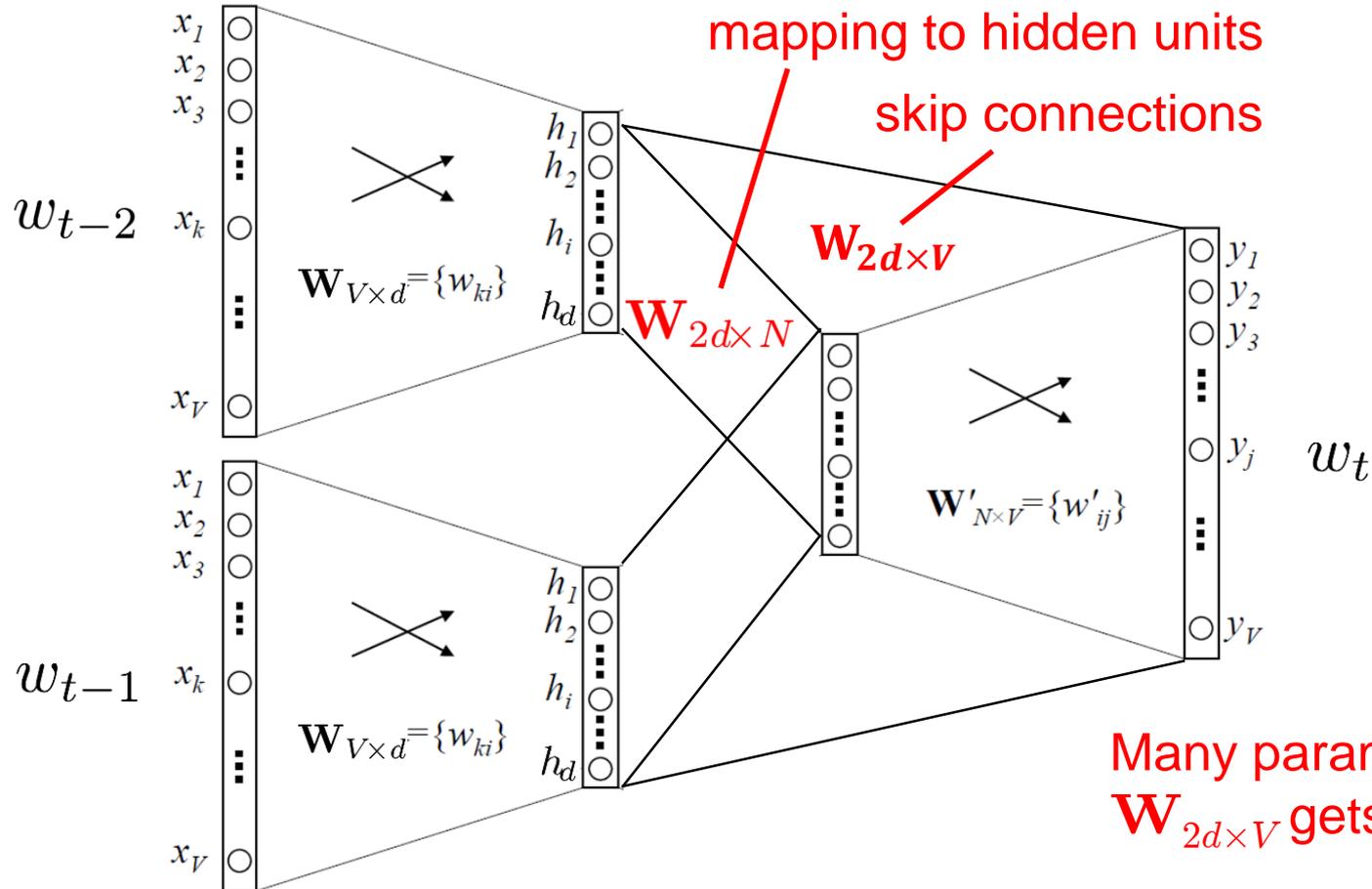
Many parameters:
$\mathbf{W}_{2d \times V}$ gets huge!

- Train on large corpus of data, learn $\mathbf{W}_{V \times d}$.

  $\Rightarrow$ Shown to outperform n-grams by [Bengio et al., 2003].

20

(part of a 2.5D map of the most common 2500 words)

Machine Learning Winter '19

Visualization of the Resulting Embedding

# Visualization of the Resulting Embedding

B. Leibe

23
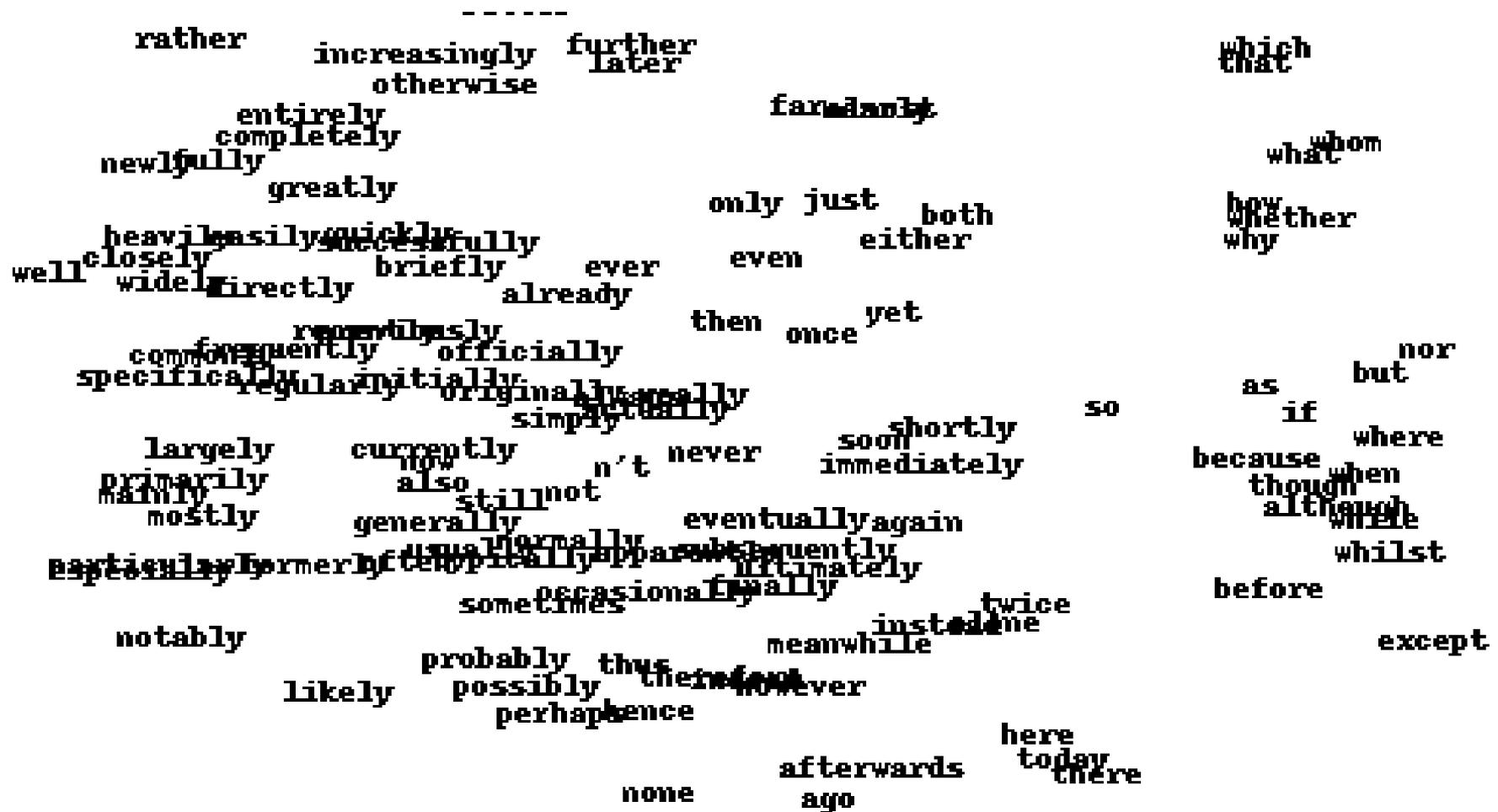
Popular Word Embeddings

- Open issue
  - *What is the best setup for learning such an embedding from large amounts of data (billions of words)?*

- Several recent improvements
  - word2vec   [Mikolov 2013]
  - GloVe   [Pennington 2014]
  - $\Rightarrow$ Pretrained embeddings available for everyone to download.

B. Leibe

word2vec

- Goal
  - Make it possible to learn high-quality word embeddings from huge data sets (billions of words in training set).

- Approach
  - Define two alternative learning tasks for learning the embedding:
    - "Continuous Bag of Words" (CBOW)
    - "Skip-gram"
  - Designed to require fewer parameters.

B. Leibe

Image source: Mikolov et al., 2015
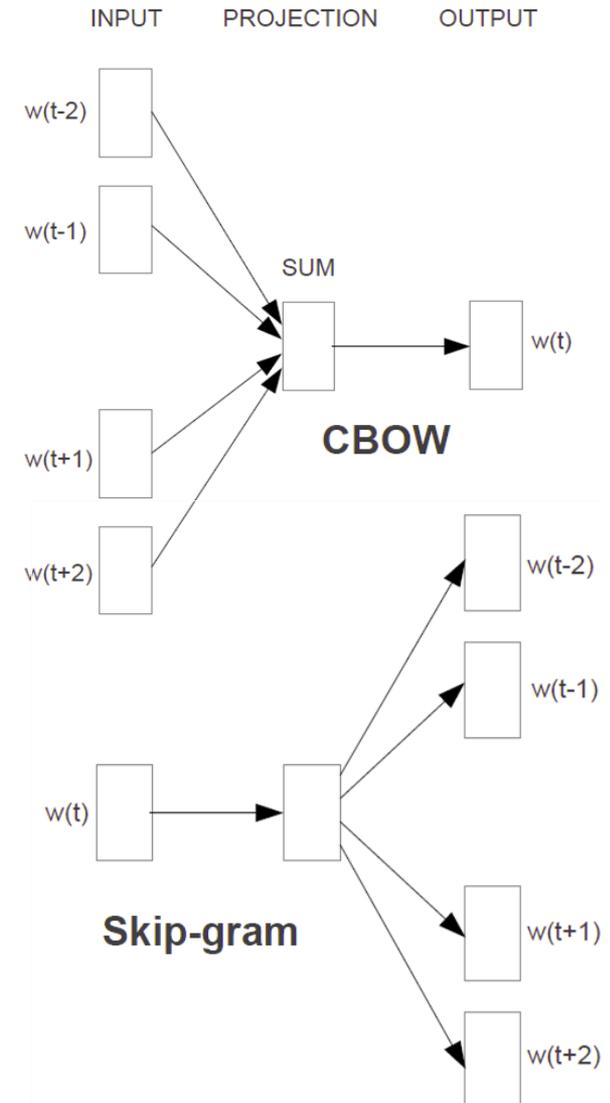
word2vec: CBOW Model

- ## Continuous BOW Model

  - ➤ Remove the non-linearity from the hidden layer

  - ➤ Share the projection layer for all words (their vectors are averaged)

  - $\Rightarrow$ Bag-of-Words model (order of the words does not matter anymore)

- ## Side note

  - ➤ Summing the encoding vectors for all words encourages the network to learn orthogonal embedding vectors for different words.

B. Leibe

Image source: Xin Rong, 2015
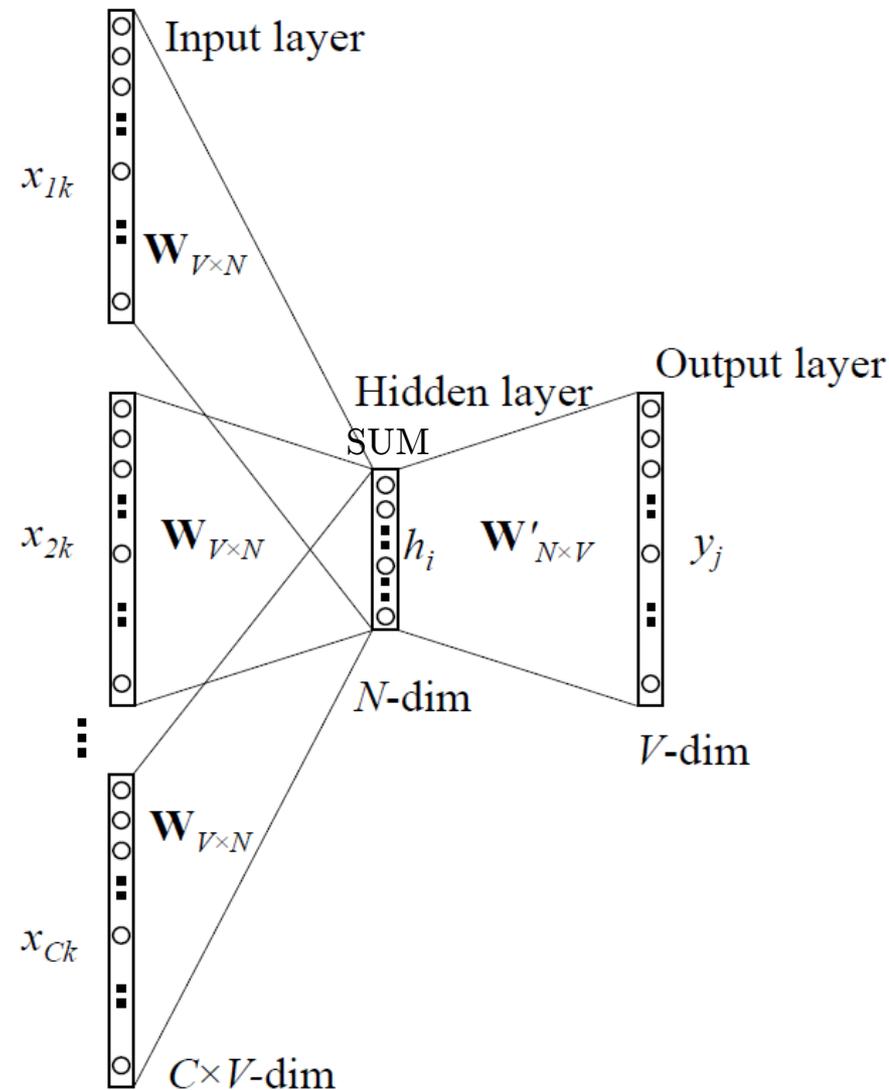
word2vec: Skip-Gram Model

- ## Continuous Skip-Gram Model
  - ➢ Similar structure to CBOW
  - ➢ Instead of predicting the current word, predict words within a certain range of the current word.
  - ➢ Give less weight to the more distant words

- ## Implementation
  - ➢ Randomly choose a number $R \in [1,C]$.
  - ➢ Use R words from history and R words from the future of the current word as correct labels.
  - $\Rightarrow R+R$ word classifications for each input.

Input layer

Hidden layer

x_k $\mathbf{W}_{V \times N}$ h_i $\mathbf{W}'_{N \times V}$

N-dim

V-dim

$\mathbf{W}'_{N \times V}$

$\mathbf{W}'_{N \times V}$

Output layer

$y_{1,j}$

$y_{2,j}$

$y_{C,j}$

$C \times V$-dim

B. Leibe

Machine Learning Winter '19
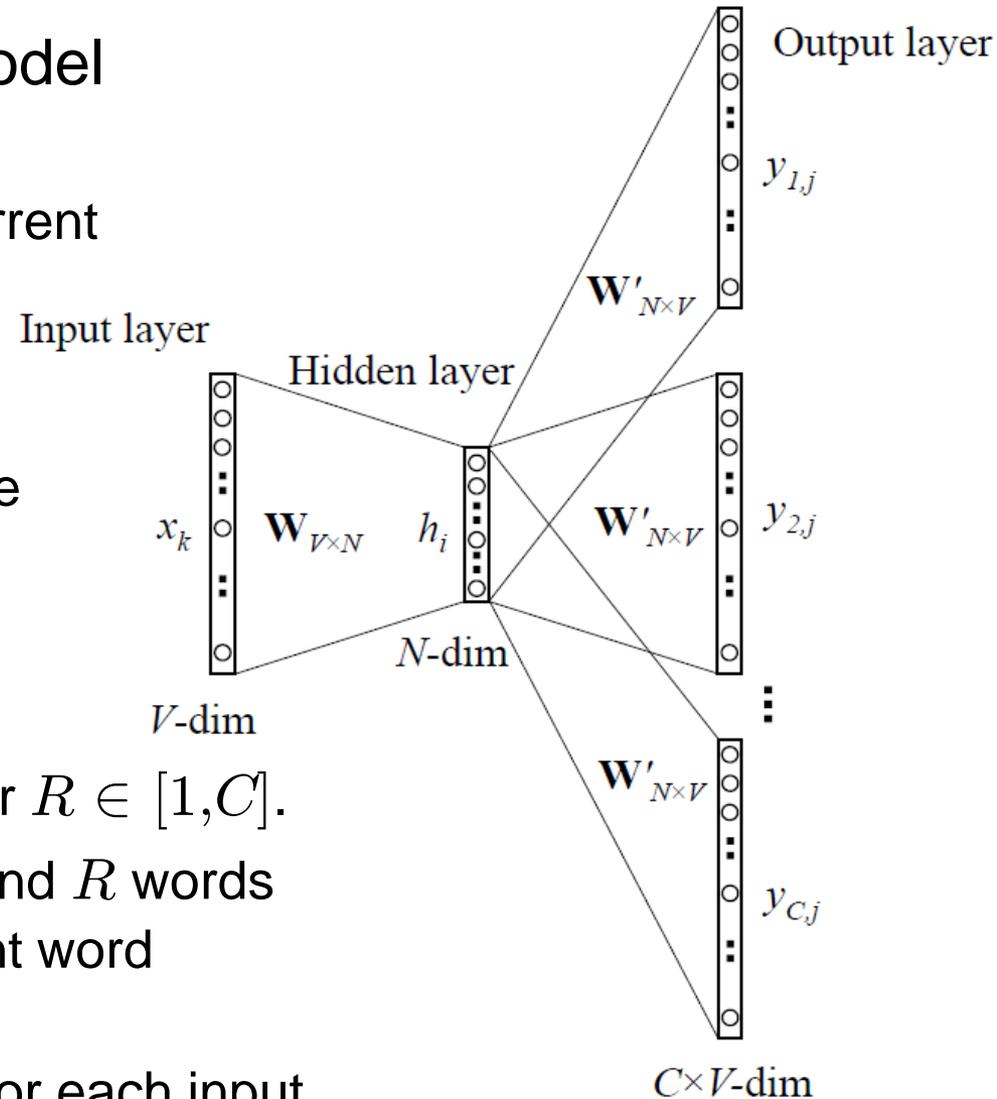
Interesting property

- Embedding often preserves linear regularities between words
  - Analogy questions can be answered through simple algebraic operations with the vector representation of words.

- Example
  - What is the word that is similar to *small* in the same sense as *bigger* is to *big*?
  - For this, we can simply compute
    X = vec(*"bigger"*) – vec(*"big"*) + vec(*"small"*)
  - Then search the vector space for the word closes to X using the cosine distance.
  - $\Rightarrow$ Result (when words are well trained): vec(*"smaller"*).

- Other example
  - E.g., vec("King") – vec("Man") + vec("Woman") $\approx$ vec("Queen")

B. Leibe

Machine Learning Winter '19

Evaluation on Analogy Questions

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

semantic (rows 1–5), *syntactic* (rows 6–14)

B. Leibe

Image source: Mikolov et al., 2015

Results

Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days x CPU cores]
			Semantic	Syntactic	Total	
NNLM	100	6B	34.2	64.5	50.8	14 x 180
CBOW	1000	6B	57.3	68.9	63.7	2 x 140
Skip-gram	1000	6B	66.1	65.1	65.6	2.5 x 125

- **Results**
  - ➢ word2vec embedding is able to correctly answer many of those analogy questions.
  - ➢ CBOW structure better for syntactic tasks
  - ➢ Skip-gram structure better for semantic tasks

B. Leibe

Image source: Mikolov et al., 2015

Machine Learning Winter '19

Problems with 100k-1M outputs

- Weight matrix gets huge!

- Example: CBOW model
  - One-hot encoding for inputs
  - $\Rightarrow$ Input-hidden connections are just vector lookups.

  - This is not the case for the hidden-output connections!
  - State h is not one-hot, and vocabulary size is 1M.
  - $\Rightarrow \mathbf{W}'_{N \times V}$ has 300$\times$1M entries
 - $\Rightarrow$ All of those need to be updated by backprop.

B. Leibe

Image source: Xin Rong, 2015
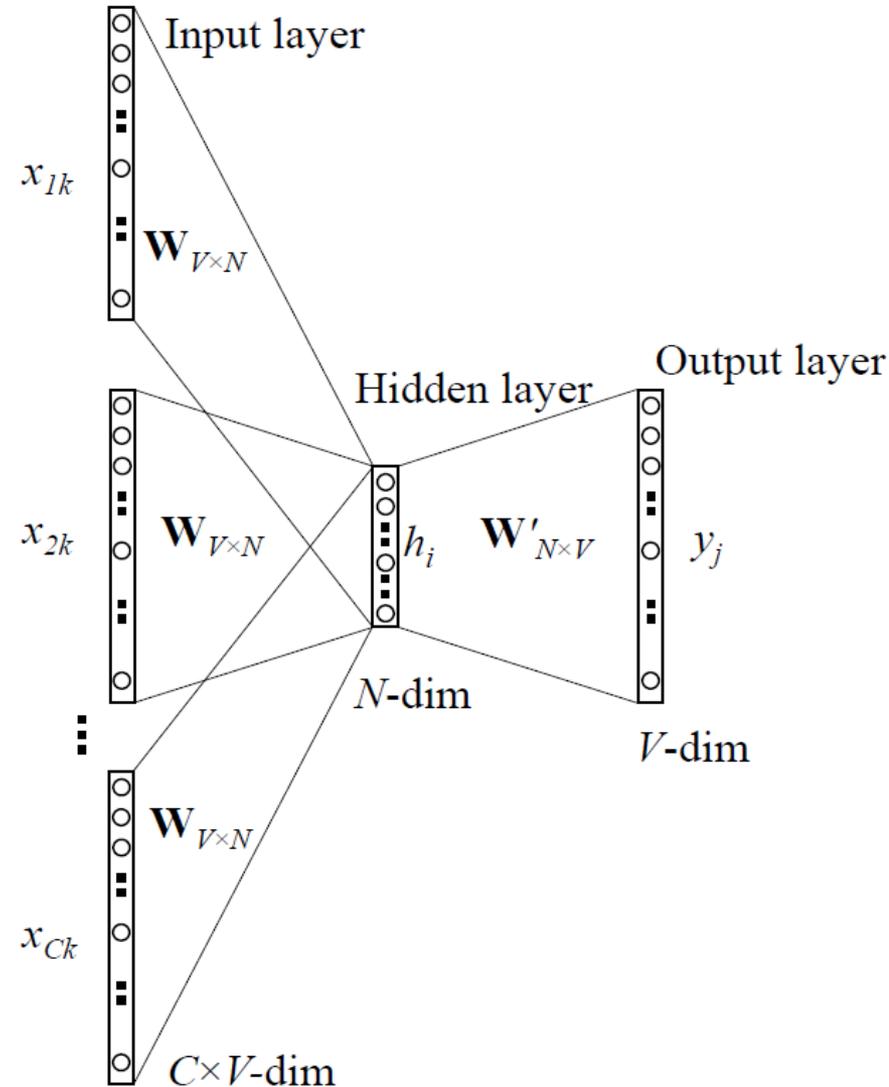
Problems with 100k-1M outputs

- **Softmax gets expensive!**
  - ➢ Need to compute normaliza-tion over 100k-1M outputs

B. Leibe
Image source: Xin Rong, 2015
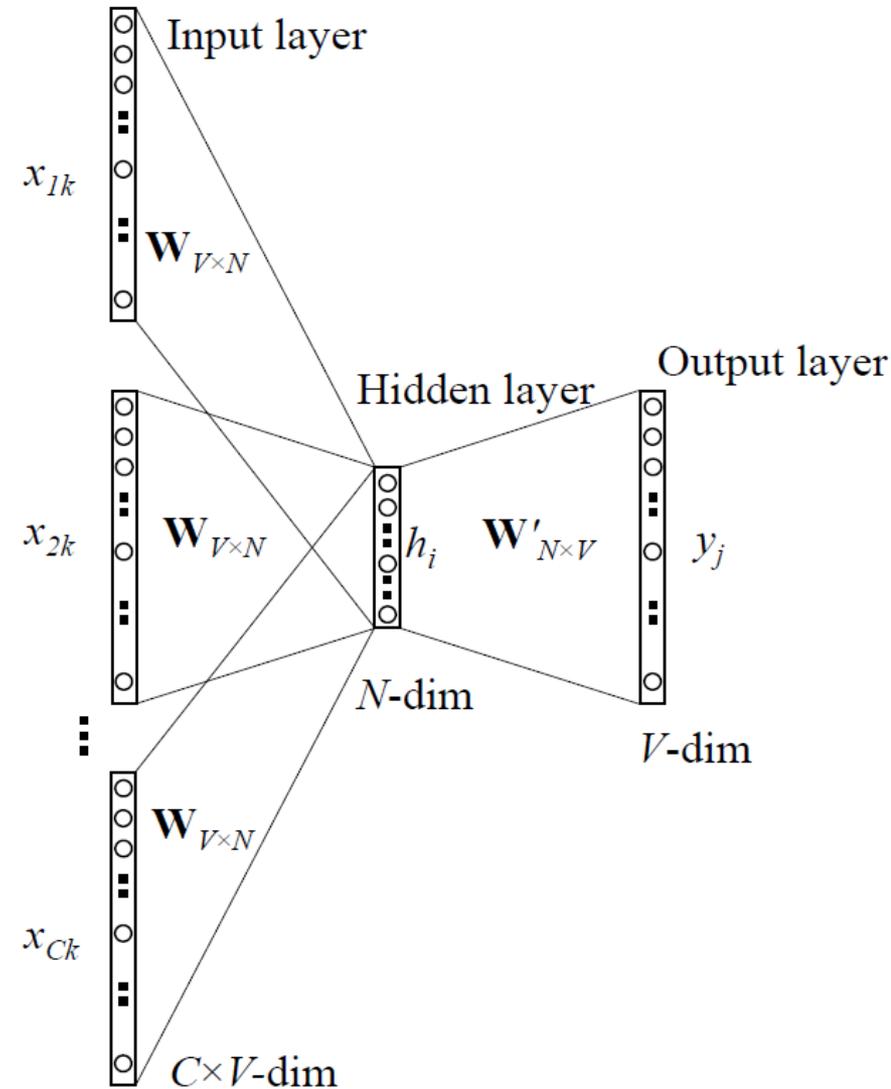
# Solution: Hierarchical Softmax
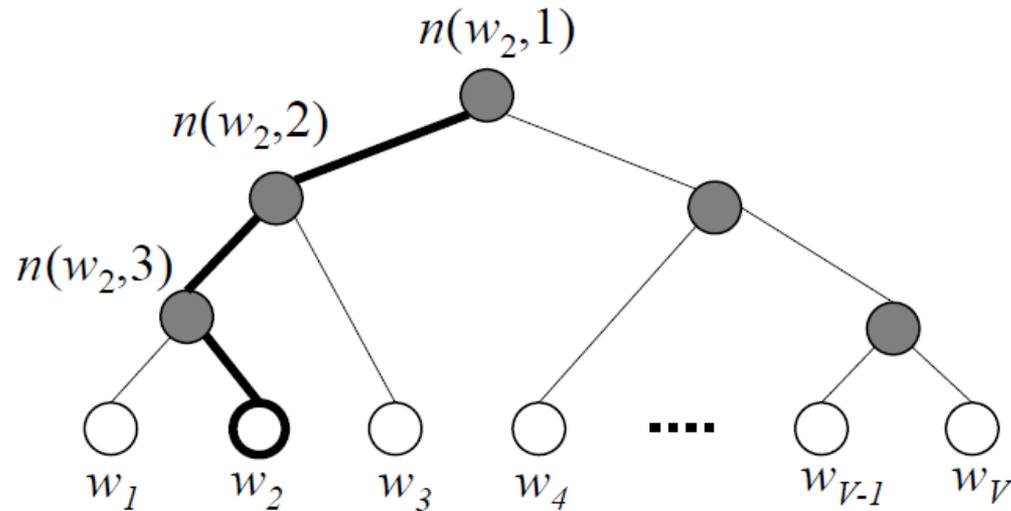
- **Idea**
  - Organize words in binary search tree, words are at leaves
  - Factorize probability of word w_0 as a product of node probabilities along the path.
  - Learn a linear decision function $y = v_{n(w,j)} \cdot h$ at each node to decide whether to proceed with left or right child node.
  - $\Rightarrow$ Decision based on output vector of hidden units directly.

B. Leibe

Topics of This Lecture

- Recap
  - ResNets
  - Applications of CNNs

- Word Embeddings
  - Neuroprobabilistic Language Models
  - word2vec
  - GloVe
  - Hierarchical Softmax

- Embeddings in Vision
  - Siamese networks
  - Triplet loss networks

- Outlook: Recurrent Neural Networks

B. Leibe

Siamese Networks
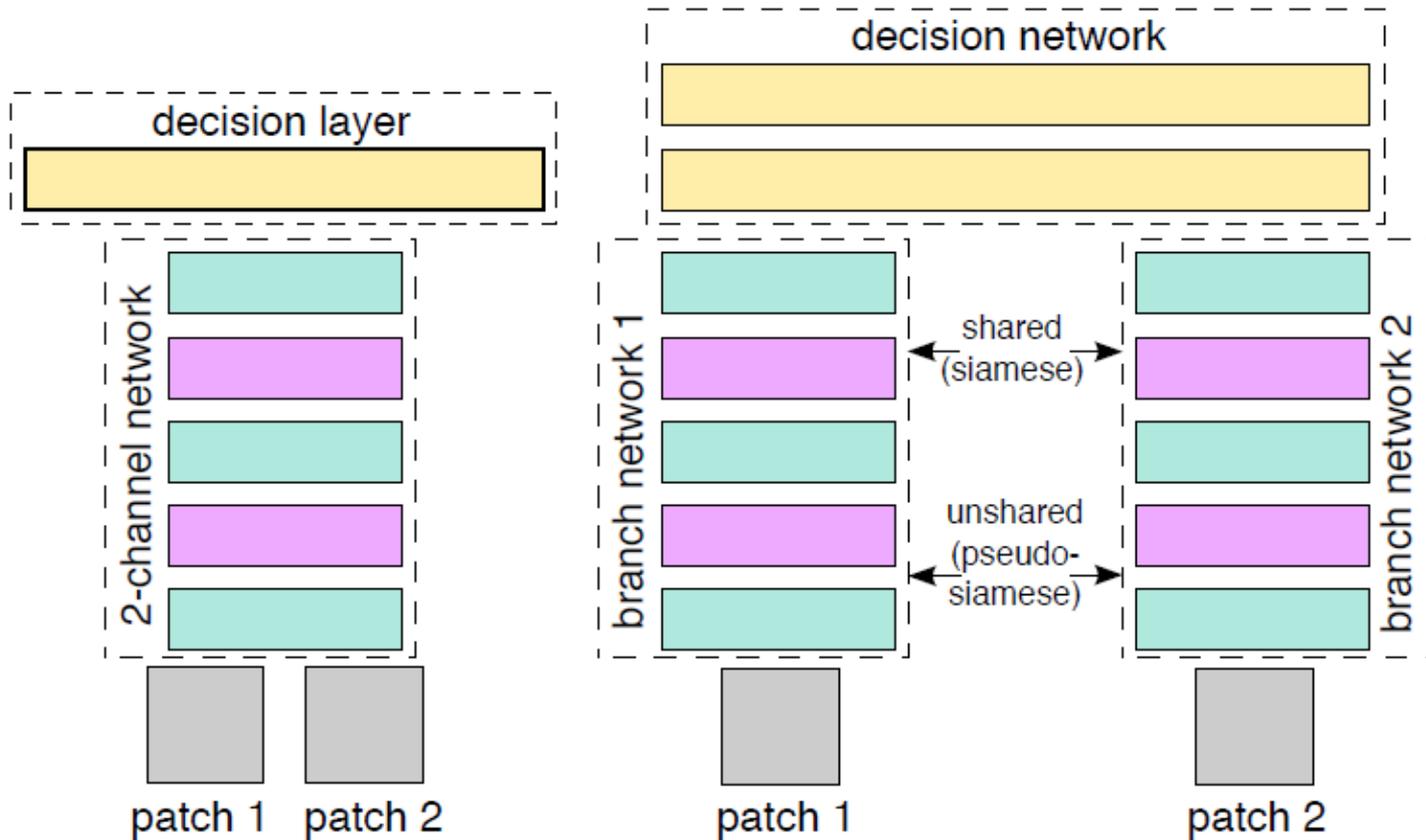
- Similar idea to word embeddings
  - Learn an embedding network that preserves (semantic) similarity between inputs
  - E.g., used for patch matching

B. Leibe

36

Discriminative Face Embeddings

- Learning an embedding using a Triplet Loss Network
 - Present the network with triplets of examples
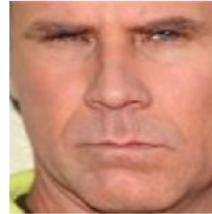
  Negative   **Anchor**   **Positive**
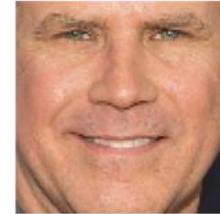
  - Apply triplet loss to learn an embedding $f(\cdot)$ that groups the positive example closer to the anchor than the negative one.

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2$$
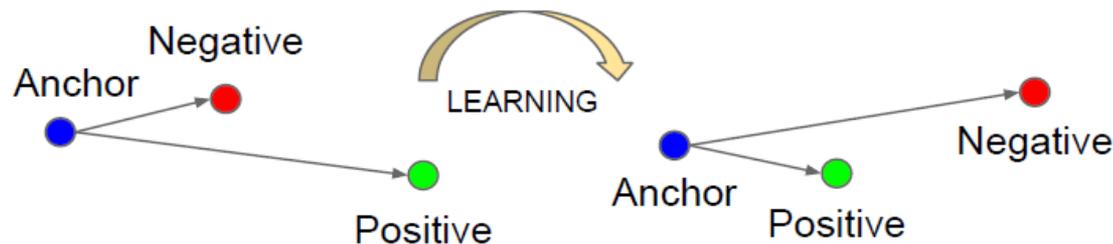
$\Rightarrow$ Used with great success in Google's FaceNet face recognition

Triplet Loss – Practical Implementation

- Triplet loss formulation

$$\mathcal{L}_{\text{tri}}(\theta) = \sum_{\substack{a,p,n \\ y_a = y_p \neq y_n}} [m + D_{a,p} - D_{a,n}]_+$$

- Practical Issue: How to select the triplets?
  - ➢ The number of possible triplets grows cubically with the dataset size.
  - ➢ Most triplets are uninformative
  - ⇒ Mining hard triplets becomes crucial for learning.
  - ⇒ Actually want *medium-hard* triplets for best training efficiency

- Popular solution: Hard triplet mining
  - ➢ Process the dataset to find hard triplets
  - ➢ Use those for learning
  - ➢ Iterate

# Triplet Loss – Practical Implementation (2)
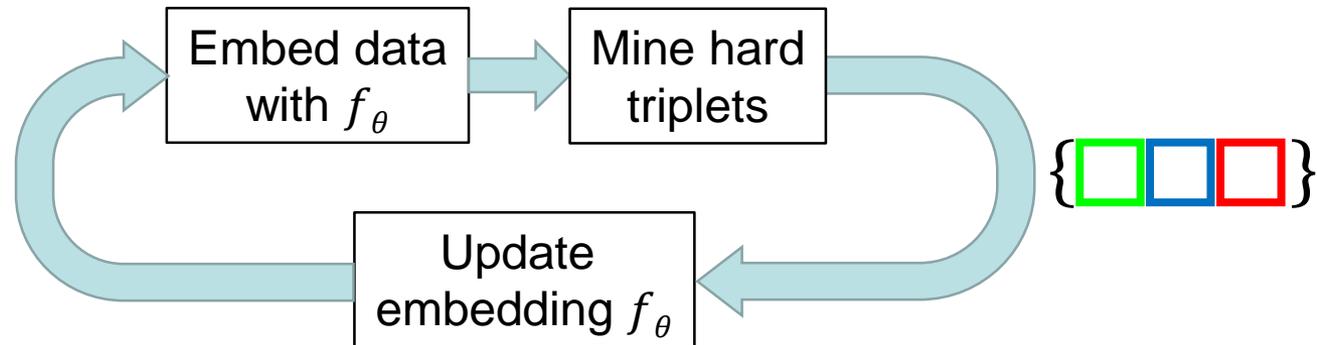
- Popular solution: Hard triplet mining
  - ➢ Process the dataset to find hard triplets
  - ➢ Use those for learning
  - ➢ Iterate

Topics of This Lecture

- Recap
  - ResNets
  - Applications of CNNs

- Word Embeddings
  - Neuroprobabilistic Language Models
  - word2vec
  - GloVe
  - Hierarchical Softmax

- Embeddings in Vision
  - Siamese networks
  - Triplet loss networks

- Outlook: Recurrent Neural Networks

B. Leibe

Outlook: Recurrent Neural Networks

| one to one | one to many | many to one | many to many | many to many |
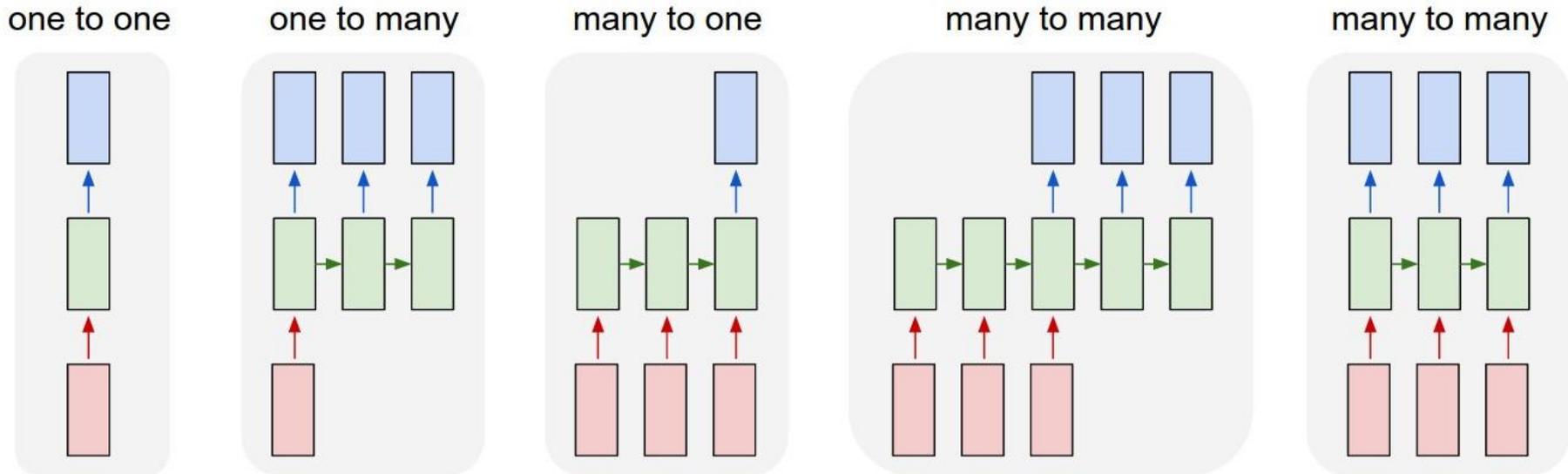
- **Up to now**
  - ➢ Simple neural network structure: 1-to-1 mapping of inputs to outputs

- **Next lecture: Recurrent Neural Networks**
  - ➢ Generalize this to arbitrary mappings

41

Image source: Andrej Karpathy

References and Further Reading

- ## Neural Probabilistic Language Model
  - Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, <u>A Neural Probabilistic Language Model</u>, In JMLR, Vol. 3, pp. 1137-1155, 2003.

- ## word2vec
  - T. Mikolov, K. Chen, G. Corrado, J. Dean, <u>Efficient Estimation of Word Representations in Vector Space</u>, ICLR'13 Workshop Proceedings, 2013.

- ## GloVe
  - Jeffrey Pennington, Richard Socher, and Christopher D. Manning, <u>GloVe: Global Vectors for Word Representation</u>, 2014.

- ## Hierarchical Softmax
  - F. Morin and Y. Bengio, <u>Hierarchical probabilistic neural network language model</u>. In AISTATS 2005.
  - A. Mnih and G.E. Hinton (2009). <u>A scalable hierarchical distributed language model</u>. In NIPS 2009.

References: Other Embeddings

- Face Embeddings
  - F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering, in CVPR 2015.
  - A. Radford, L. Metz, S. Chintala, Unsupervise Representation Learning with Deep Convolutional Generative Adversarial Networks, ICLR 2016.

B. Leibe