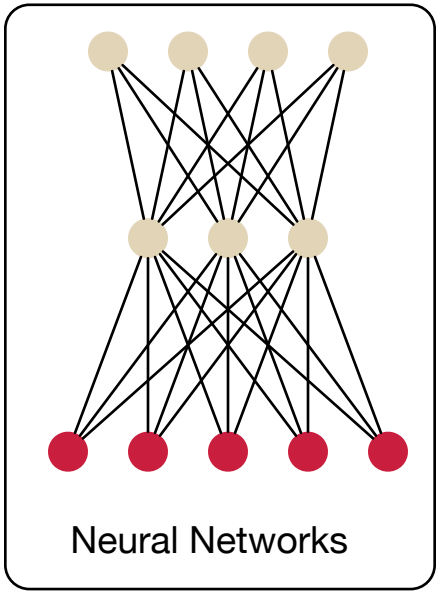
Compositionality with Deep Neural Networks

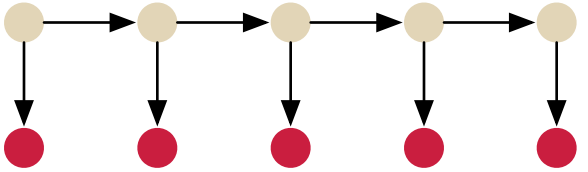
Ke Tran

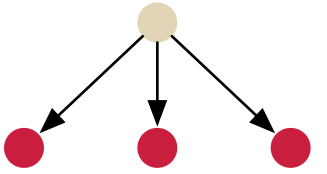
Informatics Institute
Information and Language Processing Systems

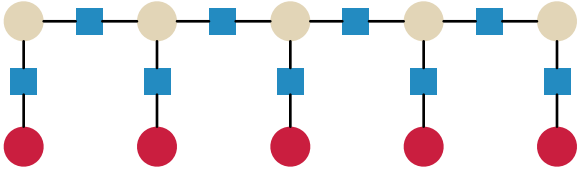

UNIVERSITY OF AMSTERDAM

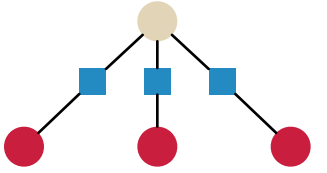
Graphical Models







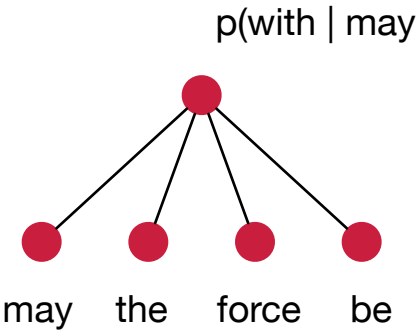




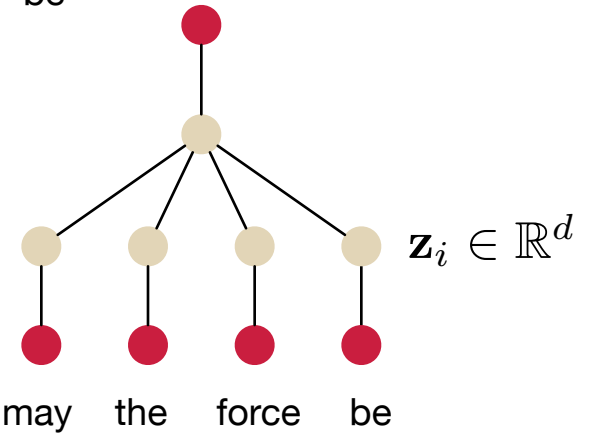
- Neural Networks are a special case of Graphical Models

Latent Variables

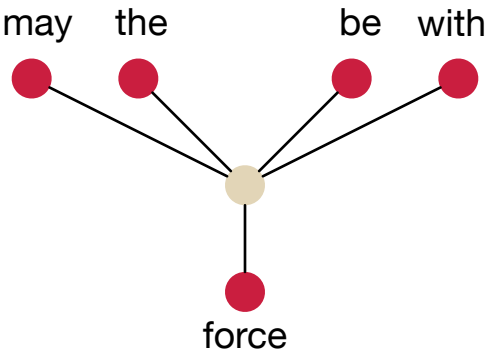
- Traditional n-gram language model



- A probabilistic neural language model
  - Each word is associated with a latent variable
  - Latent variables capture some linguistic regularizations



- Skip n-gram language model
  - Interested in learning the latent variables
  - Simplify neural LM
  - Predict surrounding words instead of the next word



Today's Lecture

- Neural Network Refresh
- Convolutional Neural Networks
- Recursive/Recurrent Neural Networks
- Auto Encoder/Recursive Encoder
- Training objectives

Supervised Learning

- Given training examples $\mathcal{D} = \{(x_i, t_i)\}_{i=1}^{N}$
  - Reviews and ratings
  - Queries and documents
  - Images and labels
- Define an appropriate model $f_{\boldsymbol{\theta}}(x)$
  - Model complexity
  - Regularization
- Define loss function to measure model predictions and ground truth
  - log-likelihood
  - MSE
- Update parameters to minimize the loss
  - update rule: taking derivative of the loss w.r.t parameters and set to zero
  - Gradient-based methods
- Perform validation to pick the best set of parameters

Softmax Classifier

- Given training examples

$$\mathcal{D} = \{(\mathbf{x}_i, t_i)\}_{i=1}^{N}$$
$$\mathbf{x}_i \in \mathbb{R}^n$$
$$t_i \in \{1, 2, \ldots, K\}$$

Task: classify unseen data to one of K possible classes

- Linear model

Parametrize each class k by a vector $\mathbf{w}_k$

Compute similarity score between input and class $f_k(\mathbf{x}_i, \mathbf{w}_k) = \mathbf{w}_k^{\mathsf{T}} \mathbf{x}_i$

Turn this score to probabilistic interpretation

$$p(t = k | \mathbf{x}_i) = \frac{\exp({\mathbf{w}_k}^{\mathsf{T}} \mathbf{x}_i)}{\sum_{j=1}^{k} \exp({\mathbf{w}_j}^{\mathsf{T}} \mathbf{x}_i)}$$

Softmax

Training: Maximizing log-likelihood of the data

$$\sum_{i=1}^{N} \ln p(t = t_i | \mathbf{x}_i) + \underbrace{R(\mathbf{w})}_{\text{regularization}}$$

Online Learning Softmax Classifier

- Goal: maximizing log-likelihood w.r.t $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K$

$$\ln p(y = k|\mathbf{x}_i) = {\mathbf{w}_k}^\mathsf{T}\mathbf{x}_i - \ln\left(\sum_{j=1}^{k} \exp({\mathbf{w}_j}^\mathsf{T}\mathbf{x}_i)\right)$$

- Taking derivatives

$$\frac{\partial}{\partial \mathbf{w}_k} \ln p(y = k|\mathbf{x}_i) = \mathbf{x}_i - \mathbf{x}_i \frac{\exp({\mathbf{w}_k}^\mathsf{T}\mathbf{x}_i)}{\sum_{j=1}^{k} \exp({\mathbf{w}_j}^\mathsf{T}\mathbf{x}_i)}$$

$$\frac{\partial}{\partial \mathbf{w}_j} \ln p(y = j|\mathbf{x}_i) = -\mathbf{x}_i \frac{\exp({\mathbf{w}_j}^\mathsf{T}\mathbf{x}_i)}{\sum_{j=1}^{k} \exp({\mathbf{w}_j}^\mathsf{T}\mathbf{x}_i)}$$

- Update parameters

$$\mathbf{w}_l^{\text{new}} = \mathbf{w}_l - \underbrace{\lambda}_{\text{learning rate}} \frac{\partial}{\partial \mathbf{w}_l} \ln p(y = l|\mathbf{x}_i)$$

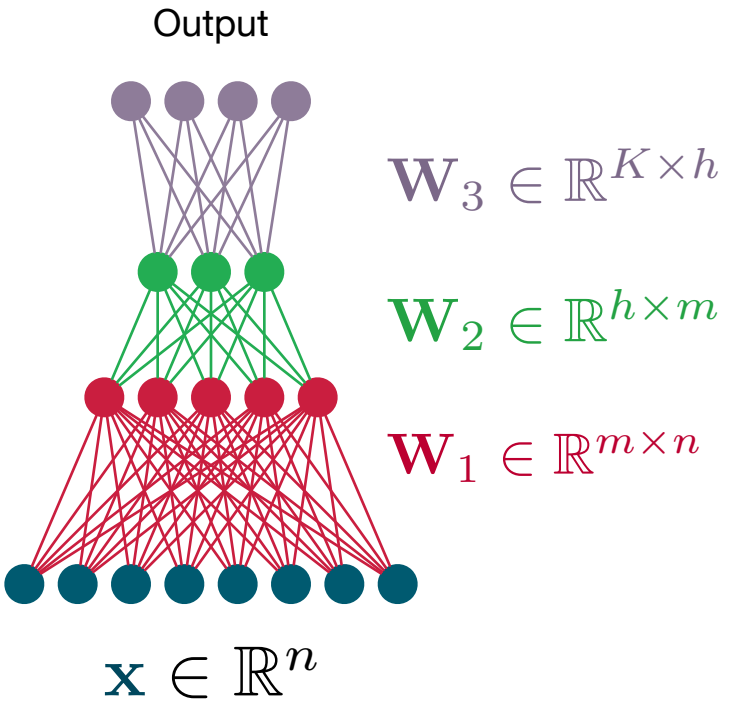
A Simple Feed-forward Neural Network

- Sigmoid activation function $\phi(z) = \frac{1}{1+\exp(-z)}$

$$\frac{\partial \phi}{\partial z} = \phi(1-\phi)$$

- Forward

$$\mathbf{y}_1 = \phi(\mathbf{W}_1\mathbf{x})$$
$$\mathbf{y}_2 = \phi(\mathbf{W}_2\mathbf{y}_1)$$
$$\mathbf{y}_3 = \phi(\mathbf{W}_3\mathbf{y}_2)$$



- Mean Square Loss

$$E = \frac{1}{2}\sum_{j\in \text{output}}(t^j - y_3^j)^2$$

or preferably in vector form $E = \frac{1}{2}(\mathbf{t}-\mathbf{y}_3)^\mathsf{T}(\mathbf{t}-\mathbf{y}_3)$

Backpropagation Vectorization

- We want to compute

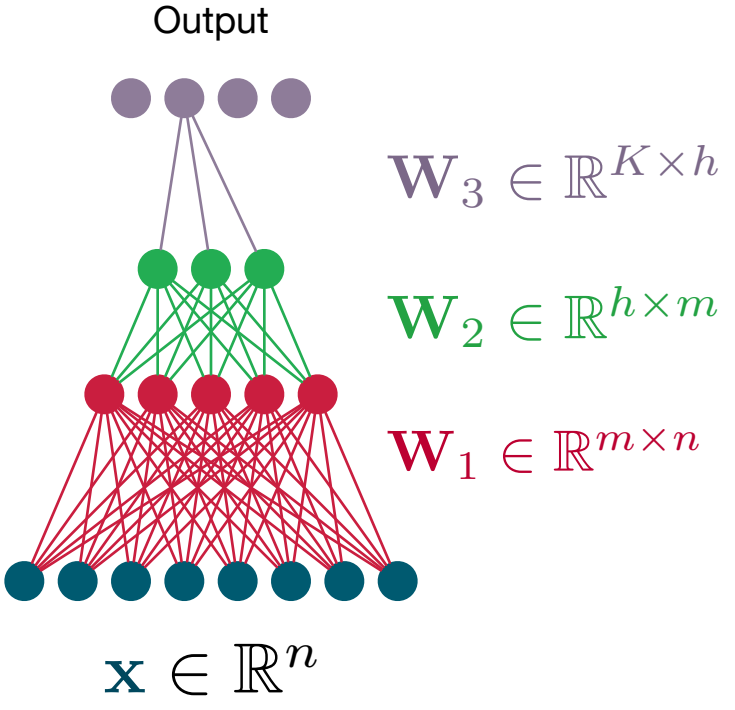
$$\frac{\partial E}{\partial \mathbf{W}_1} \qquad \frac{\partial E}{\partial \mathbf{W}_2} \qquad \frac{\partial E}{\partial \mathbf{W}_3}$$

- Backpropagation

$$\frac{\partial E}{\partial \mathbf{W}_3^j} = \frac{\partial E}{\partial \mathbf{y}_3^j} \frac{\partial \mathbf{y}_3^j}{\partial \mathbf{W}_3^j}$$

$$\frac{\partial E}{\partial \mathbf{y}_3^j} = -(\mathbf{t}^j - \mathbf{y}_3^j)$$

$$\frac{\partial \mathbf{y}_3^j}{\partial \mathbf{W}_3^j} = \phi(1-\phi)(\mathbf{W}_3^j \mathbf{y}_2)\mathbf{y}_2$$



Backpropagation vectorization

- We want to compute

$$\frac{\partial E}{\partial \mathbf{W}_1} \qquad \frac{\partial E}{\partial \mathbf{W}_2} \qquad \frac{\partial E}{\partial \mathbf{W}_3}$$

- Backprop

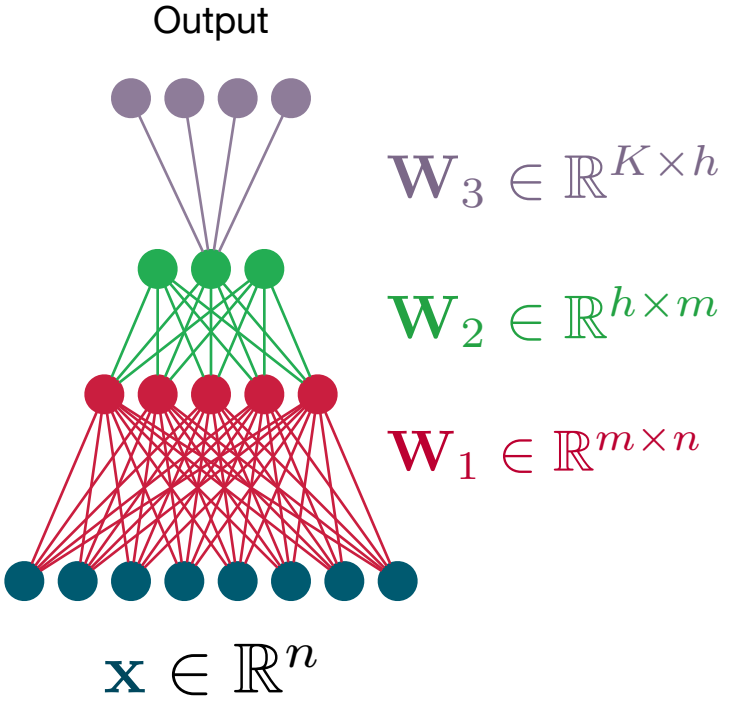
$$\frac{\partial E}{\partial \mathbf{W}_2^j} = \frac{\partial E}{\partial \mathbf{y}_2^j} \frac{\partial \mathbf{y}_2^j}{\partial \mathbf{W}_2^j}$$

$$\frac{\partial E}{\partial \mathbf{y}_2^j} = \frac{\partial E}{\partial \mathbf{y}_3} \frac{\partial \mathbf{y}_3}{\partial \mathbf{y}_2^j}$$

$$\frac{\partial E}{\partial \mathbf{y}_3} = -(\mathbf{t} - \mathbf{y}_3)$$

$$\frac{\partial \mathbf{y}_3}{\partial \mathbf{y}_2^j} = \mathbf{W}_3^{[:j]}$$

$$\frac{\partial \mathbf{y}_2^j}{\partial \mathbf{W}_2^j} = \phi(1-\phi)(\mathbf{W}_2^j \mathbf{y}_1)\mathbf{y}_1$$



Optimizations

- We want to optimize objective function $f(\mathbf{x}, \mathbf{t}, \boldsymbol{\theta})$ w.r.t parameters $\boldsymbol{\theta}$

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} f(\mathbf{x}, \mathbf{t}, \boldsymbol{\theta})$$

- Gradient-based Methods

$f(\mathbf{x}, \mathbf{t}, \boldsymbol{\theta})$

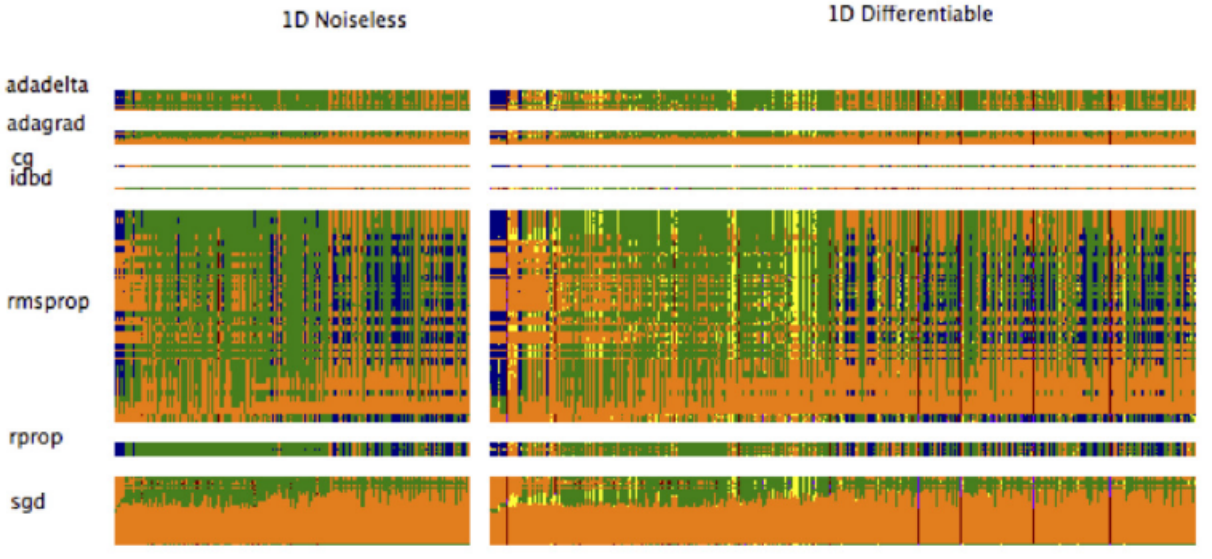
$\frac{\partial}{\partial \boldsymbol{\theta}} f(\mathbf{x}, \mathbf{t}, \boldsymbol{\theta})$

Optim

- Lots of optimization tools
  - SGD
  - L-BFGS
  - AdaGrad
  - AdaDelta
  - RMSProp

Optimizations

Unit Tests for Stochastic Optimization
Tom Schaul, Ioannis Antonoglou, David Silver
arXiv:1312.6055

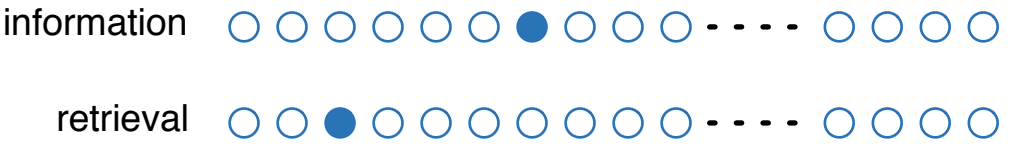


red/violet=divergence, orange=slow, yellow=variability, green=acceptable, blue=excellent

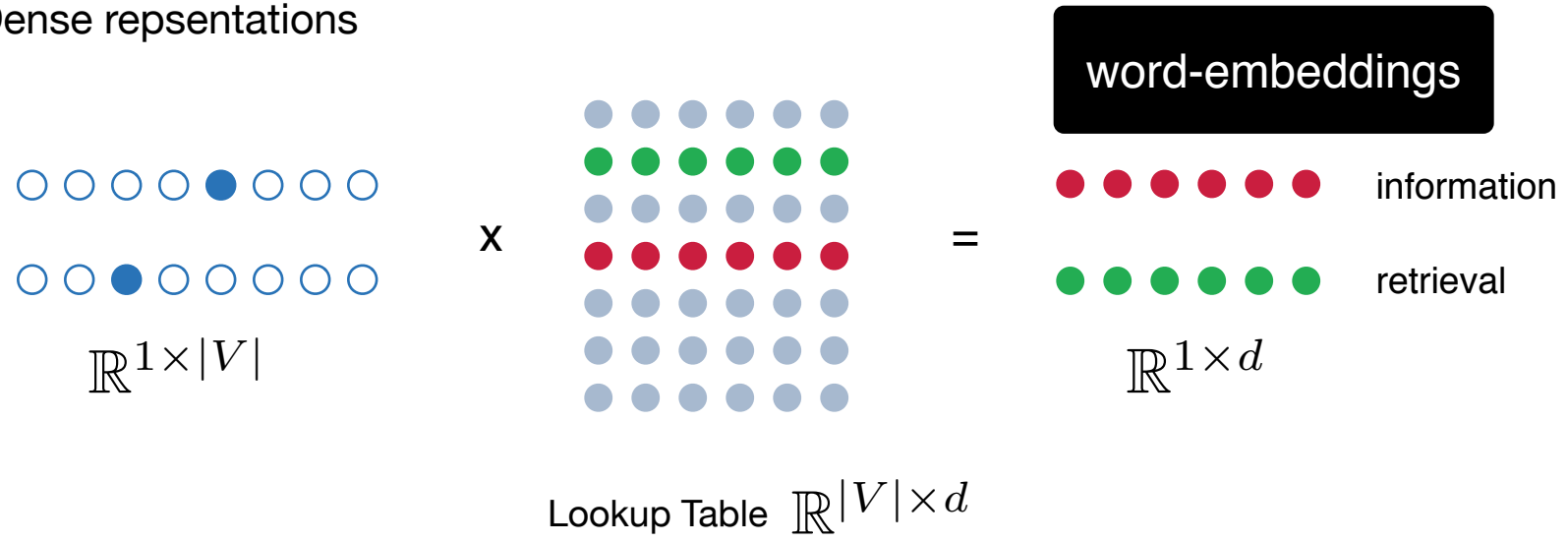
- Ada delta and RMSProp work better than others (Adagrad, SGD, …)

Dense Representations

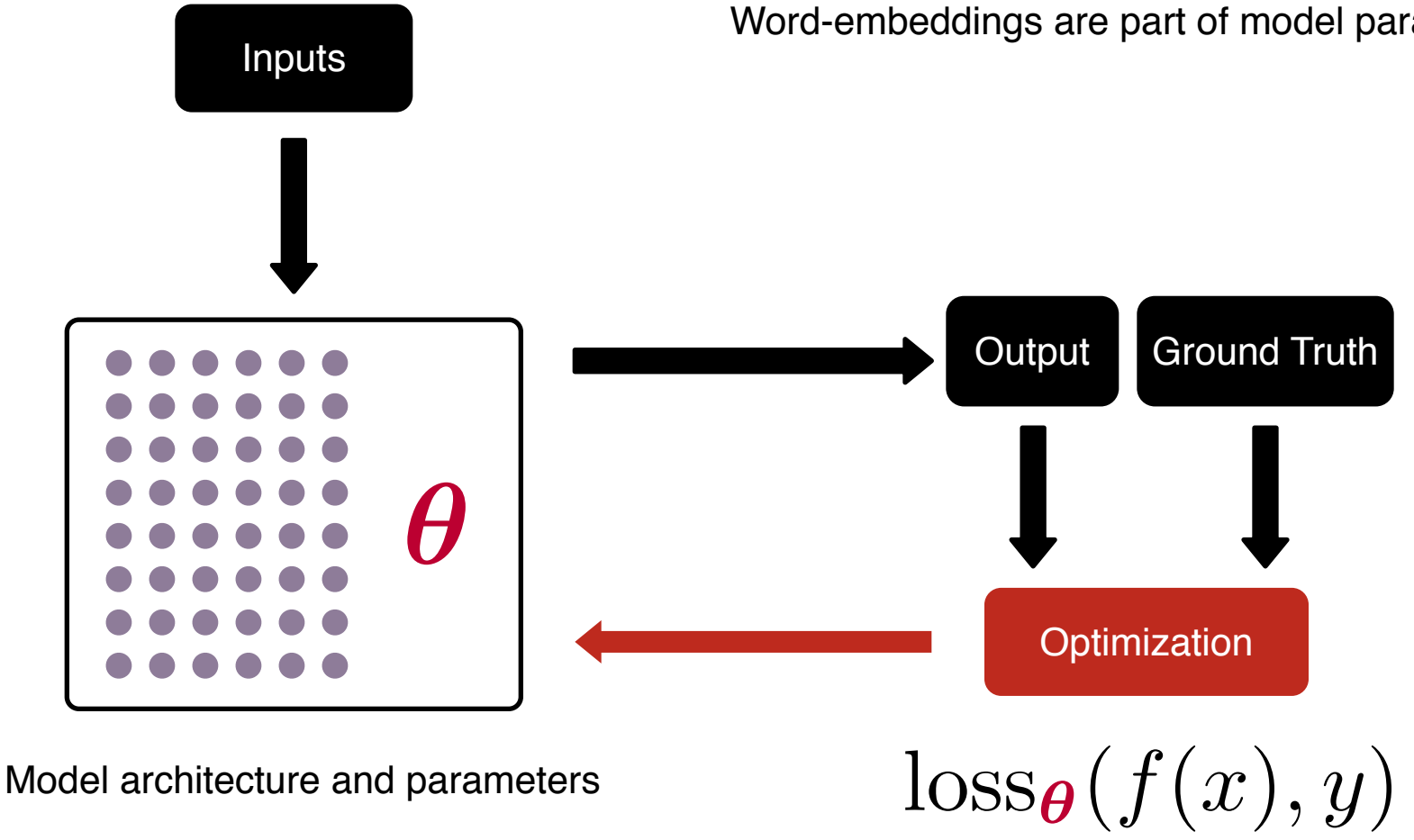
- One hot representations (or 1 of |V| representations)



- Dense repsentations

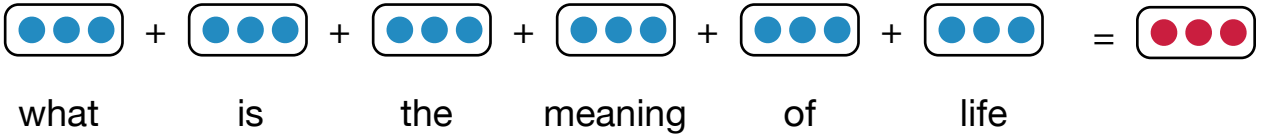


Learning Pipeline

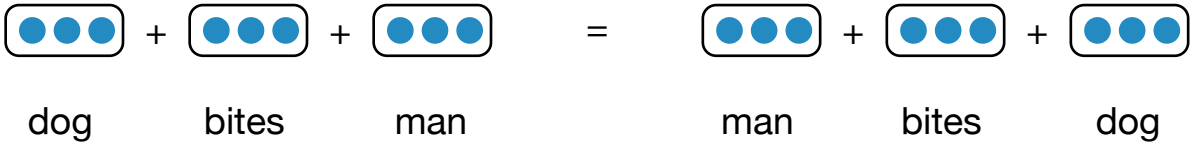


Bag of Words Model

- what is the meaning of life?



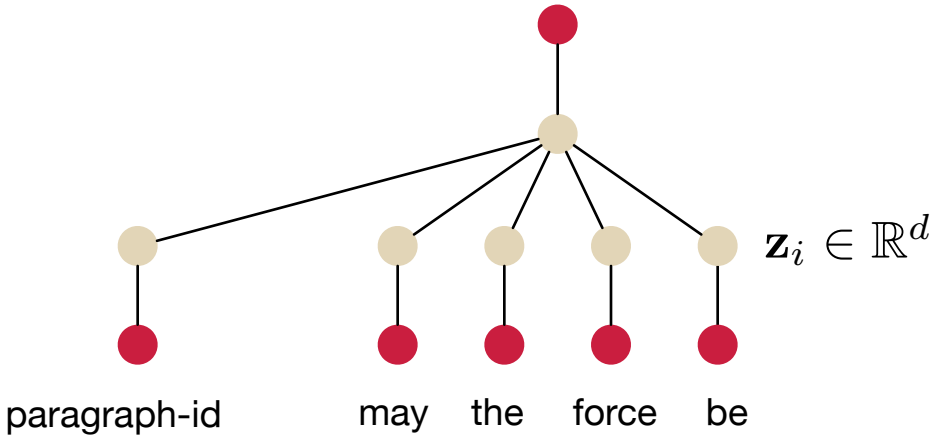
- Not very sensitive



- We need to account for the structure of sentences

Google paragraph vector

Distributed Representations of Sentences and Documents.
Quoc Le and Tomas Mikolov. ICML 2014.

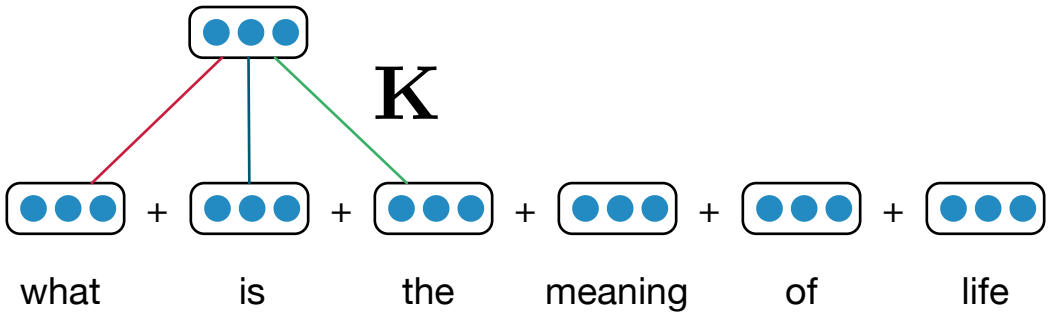


- a simple extension of word2vec

- p(word | surrounding words, paragraph)

Convolutional Neural Networks

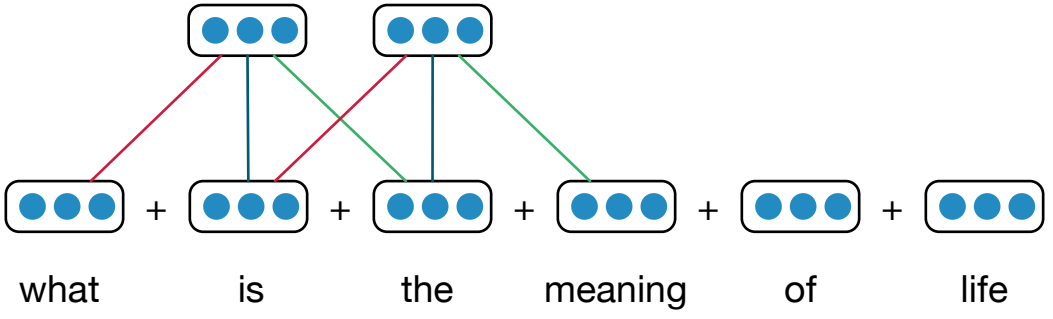
- Local n-gram model



Convolutional Neural Networks

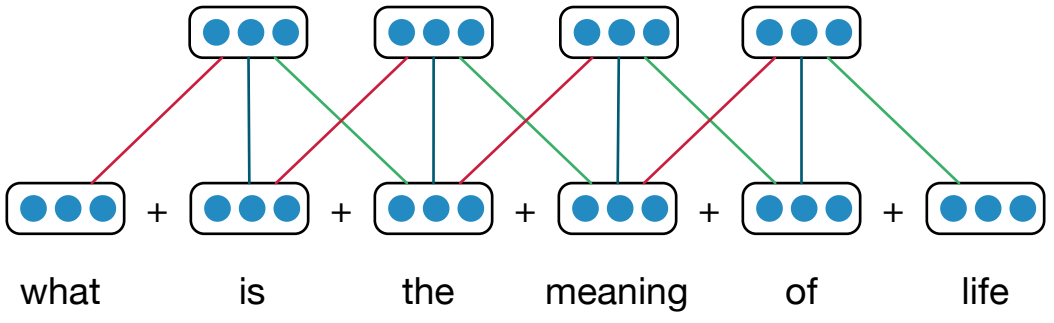
- Local n-gram model

Shared weight matrix $\mathbf{K}$



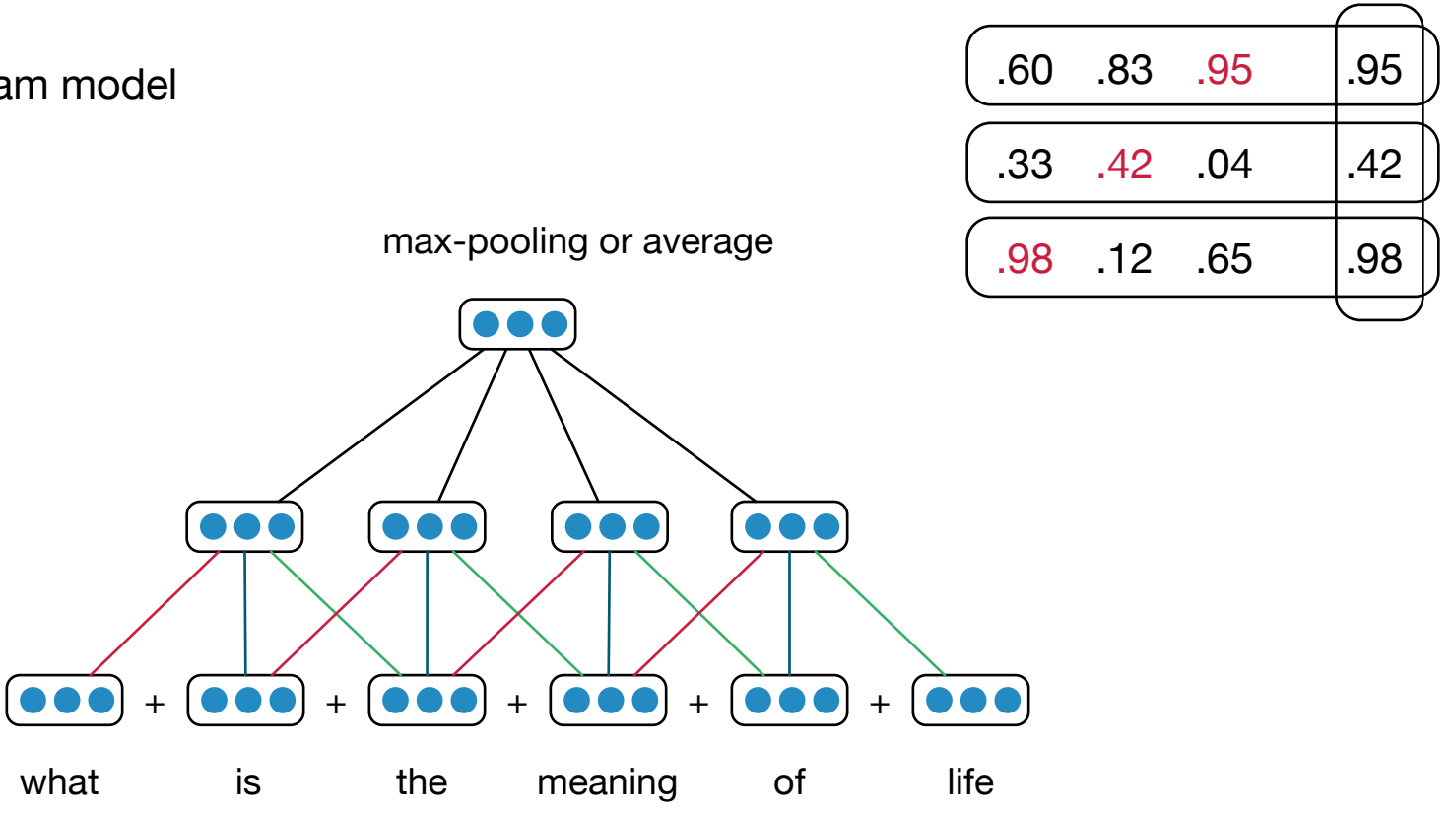
Convolutional Neural Networks

- Local n-gram model



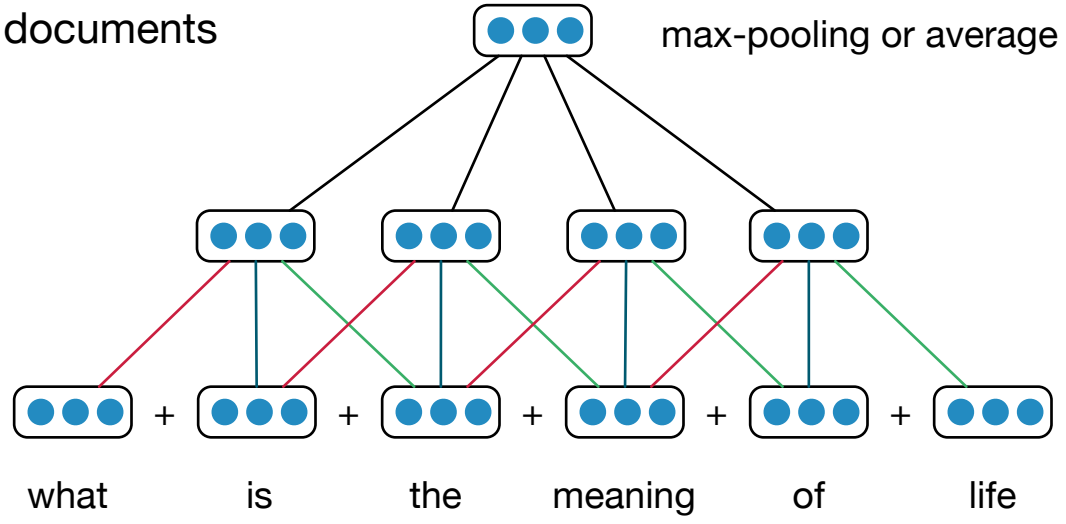
Convolutional Neural Networks

- Local n-gram model



Convolutional Neural Networks

- Summarize sentences, paragraphs, documents

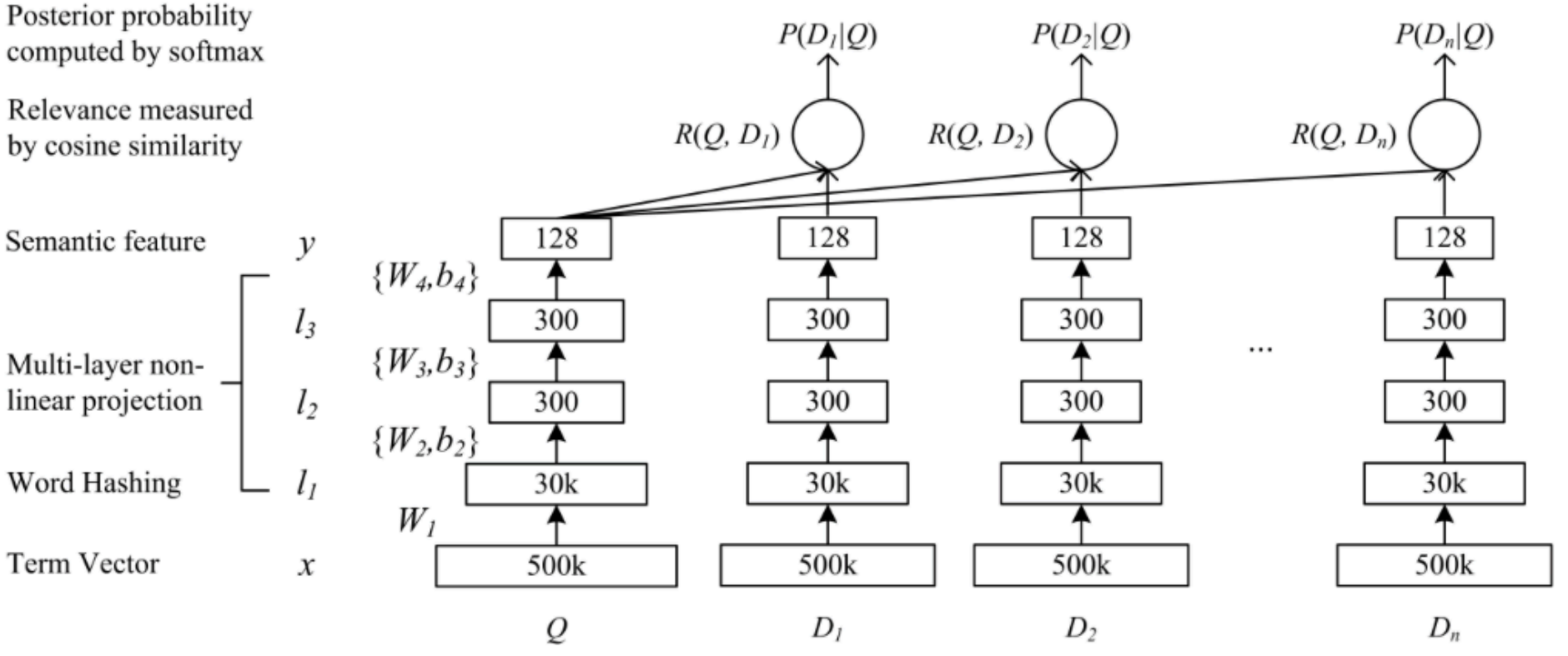


- Dynamically determine the number of convolutional layers

Recurrent Convolutional Neural Networs for Discourse Compositionality
Nal Kalchbrenner and Phil Blunsom. ACL 2013

Semantic Matching

- Match query and documents in latent space



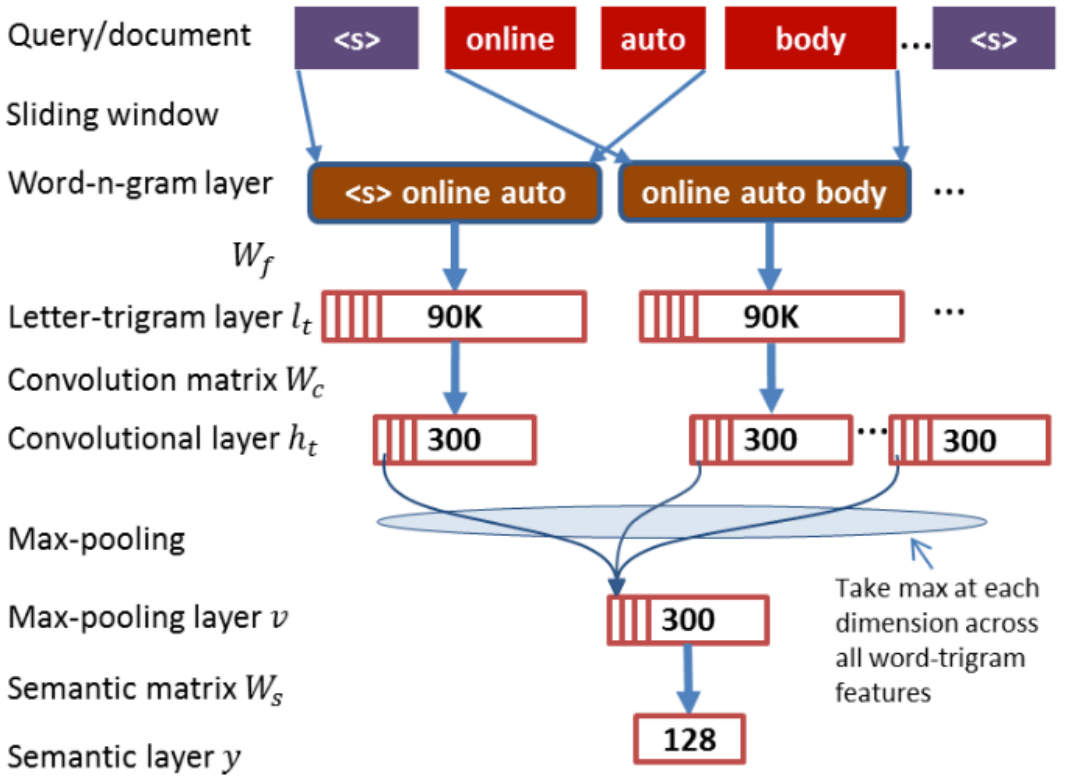
Learning deep structured semantic models for web search using clickthrough data
Po-Sen Huang et. al. CIKM 2013

Semantic Matching

- Word-Hashing
  - The size of vocabulary is very large in web search.
  - Word-Embeddings are unreliable for rare words
  - Why not n-gram character?

Vocabulary Size	# letter-trigrams	# collisons
40,000	10,306	2
500,000	30,621	22

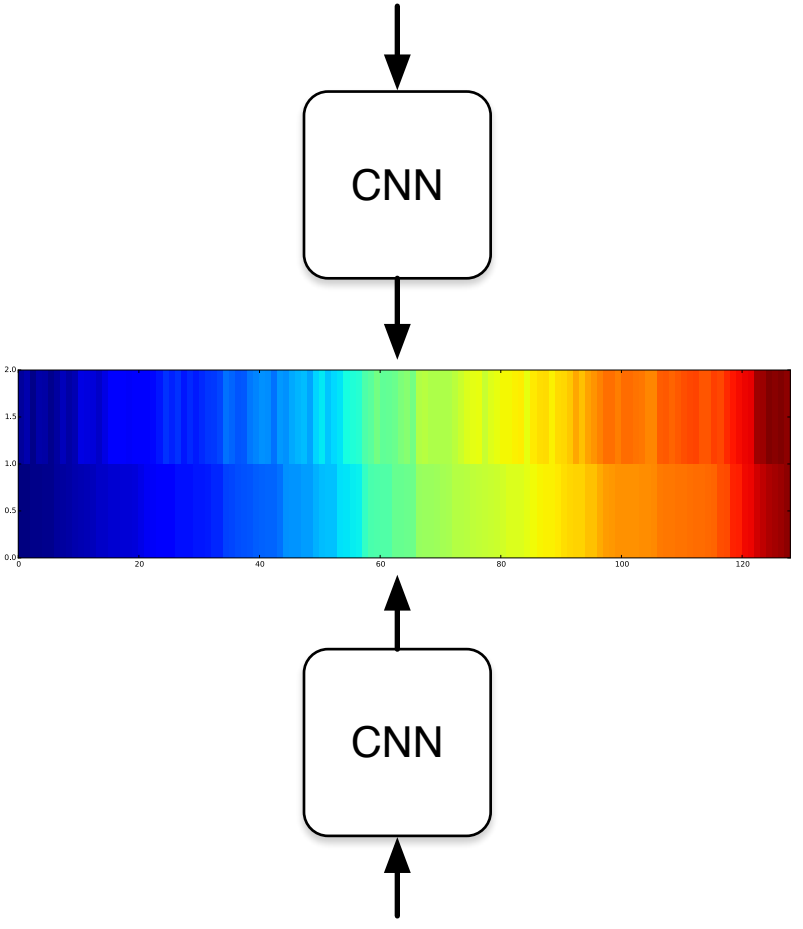
Semantic Matching



Convolutional Latent Semantic Model (CLSM) learns vector representations for word-ngrams in query and document using a DSSM-like architecture, and combines them using max-pooling.

Semantic Matching

What happens if our body absorbs excessive amount vitamin d



calcium supplements and vitamin d discussion stop sarcoidosis

Results

C

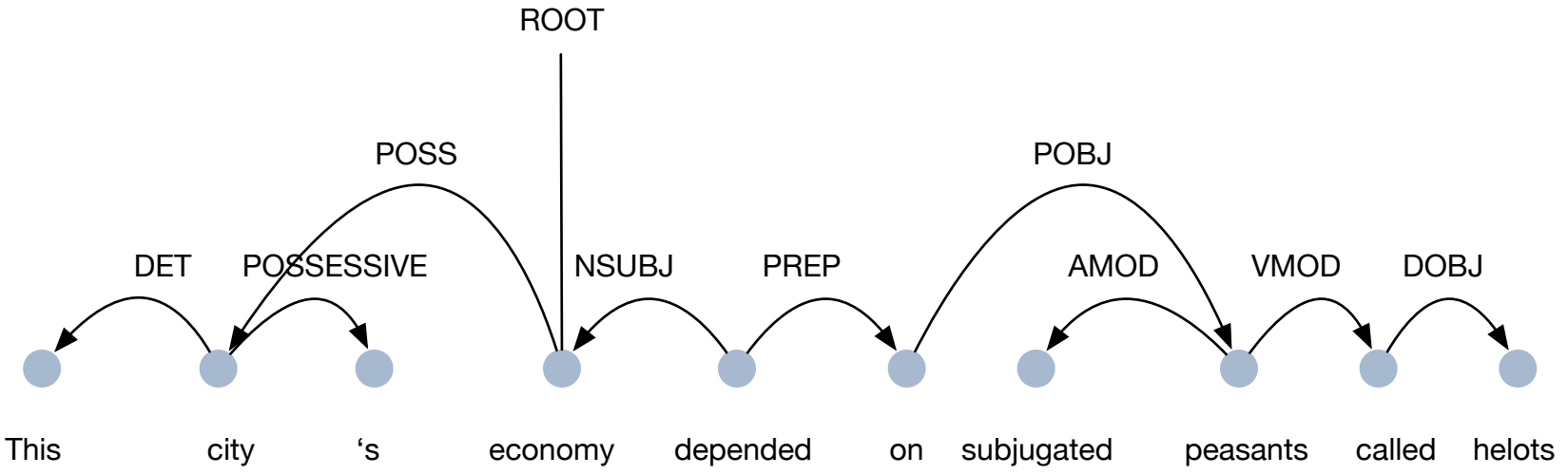
Model	NDCG@1	NDCG@3	NDCG@10
BM25 (baseline)	0.308	0.373	0.455
Latent Semantic Model	0.298	0.372	0.455
Probabilistic Latent Semantic Model	0.295	0.371	0.456
Word-based Translation Model	0.332	0.400	0.478
Bilingual Translation Model	0.337	0.403	0.480
Deep Structured Semantic Model	0.362	0.425	0.498

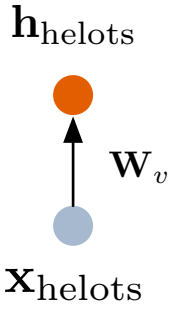
- Compositionally makes a difference

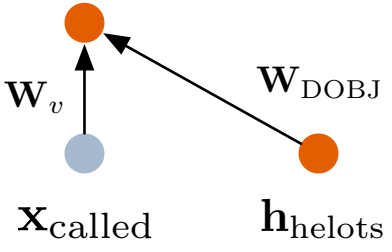
Hinton AMA http://www.reddit.com/r/MachineLearning/comments/2lmo0l/ama_geoffrey_hinton

What's wrong with CNN: http://techtv.mit.edu/collections/bcs/videos/30698-what-s-wrong-with-convolutional-nets

Recursive Neural Networks





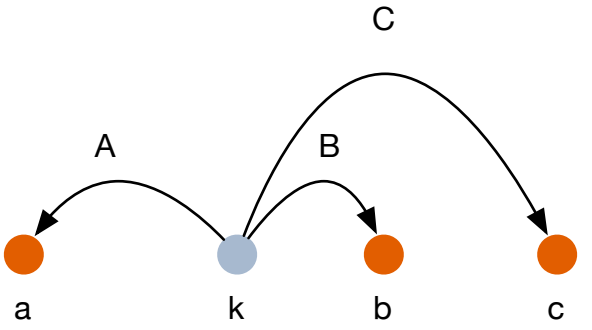


$$\mathbf{h}_{\text{helots}} = f(\mathbf{W}_v \mathbf{x}_{\text{helots}} + \mathbf{b})$$

$$\mathbf{h}_{\text{called}} = f(\mathbf{W}_v \mathbf{x}_{\text{called}} + \mathbf{W}_{\text{DOBJ}} \mathbf{h}_{\text{helots}} + \mathbf{b})$$

Recursive Neural Networks

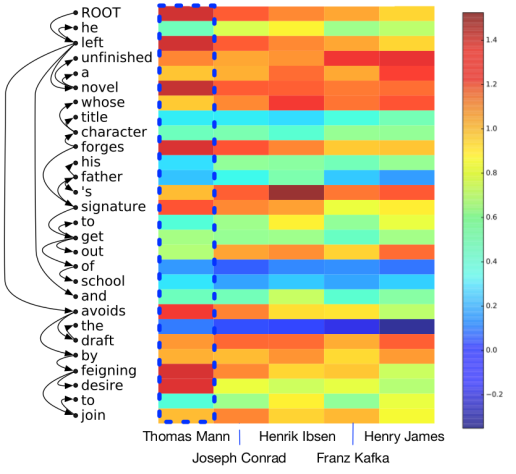
- a,b,c,x are words



- A,B,C are dependency relations

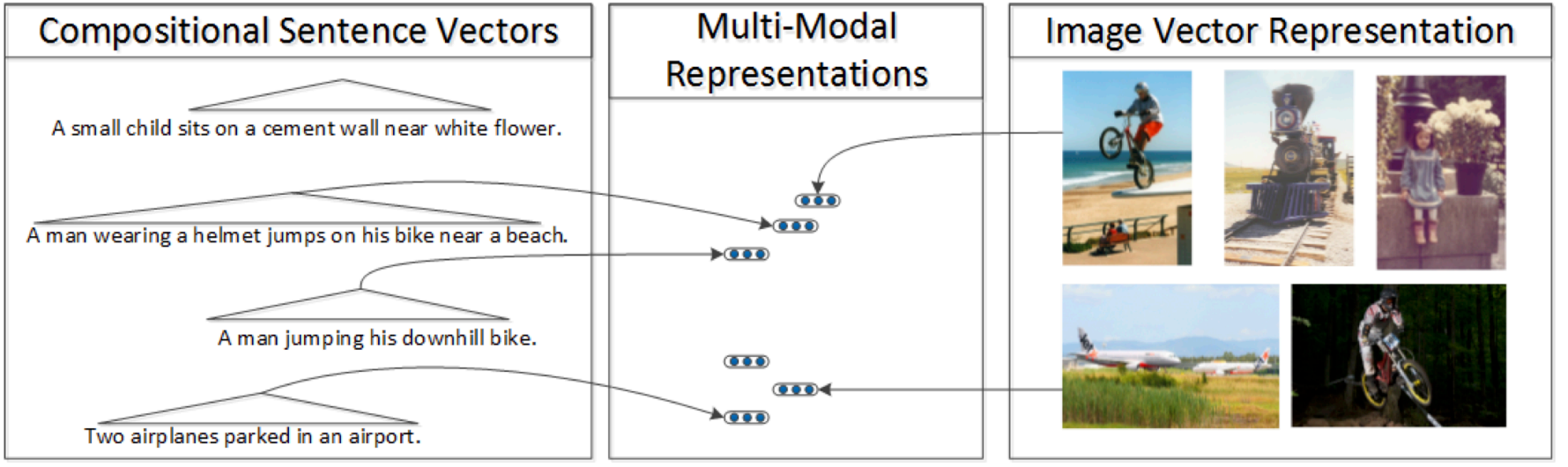
- $\mathbf{h}_k = f(\mathbf{W}_v \mathbf{x}_k + \mathbf{W}_a \mathbf{h}_a + \mathbf{W}_b \mathbf{h}_b + \mathbf{W}_c \mathbf{h}_c + \mathbf{b})$

- Softmax at every node of the tree



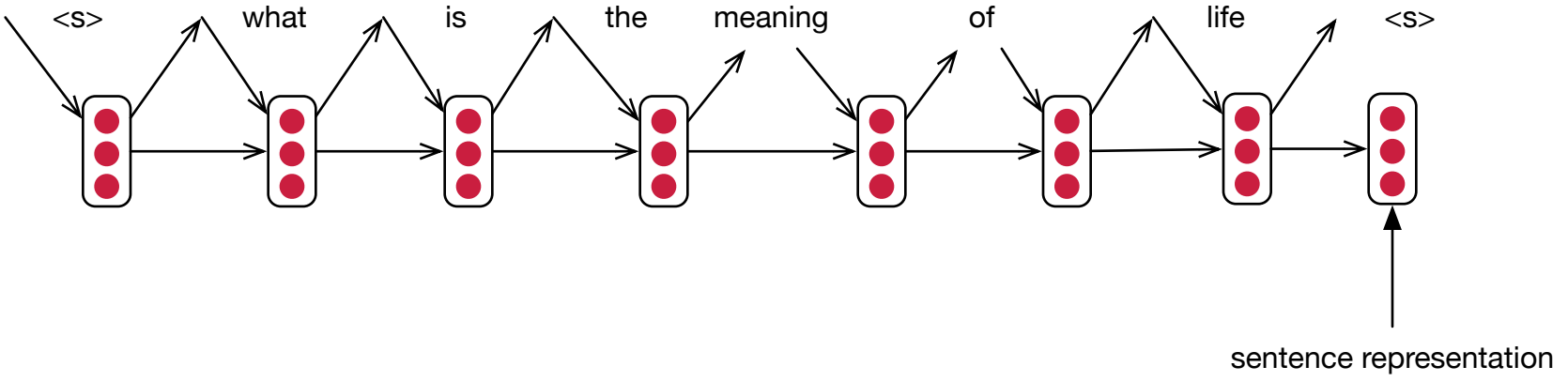
A Neural Network for Factoid Question Answering over Paragraphs
Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III
EMNLP 2014

Recursive Neural Networks



Grounded Compositional Semantics for Finding and Describing Images with Sentence.
Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, Andrew Y. Ng.
Transactions of the Association for Computational Linguistics (TACL 2014), Presented at ACL 2014.

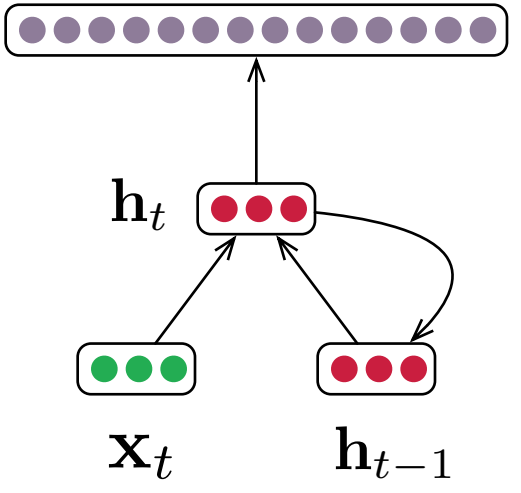
Recurrent Neural Networks



- Predict the next word given the whole unbounded history

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{V}\mathbf{h}_{t-1})$$

$$\mathbf{y}_t = \text{softmax}(\mathbf{H}\mathbf{h}_t + \mathbf{b})$$

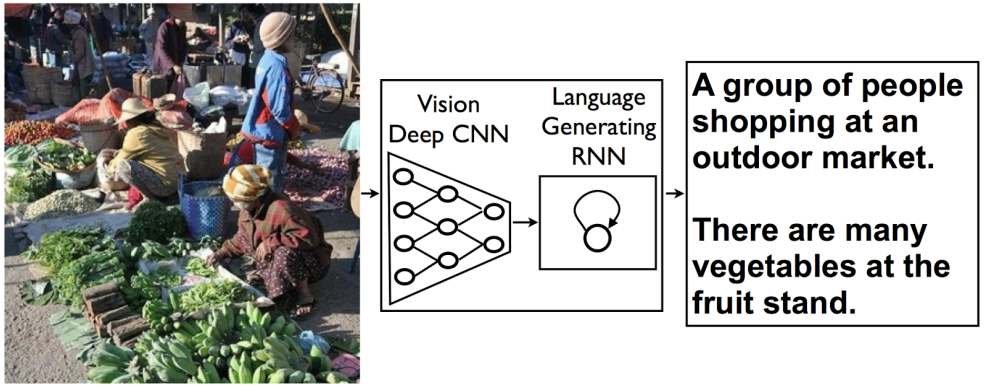


- Long Short Term Memory (LSTM)

Recurrent Neural Networks

- Generate text describing images

Show and Tell: A Neural Image Caption Generator.
Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan
arXiv:1411.4555.



- Encode source sentence with RNN and decode with RNN

Sequence to Sequence Learning with Neural Networks.
Ilya Sutskever, Oriol Vinyals, Quoc Le, NIPS 2014

- Parsing with RNN

Grammar as Foreign Language
Oriol Vinyals et al.
arXiv 2014

Auto Encoder

- Map data to lower dimension

$$\mathbf{z} = \phi(\mathbf{W}_1^{\mathsf{T}}\mathbf{x})$$

- Force model to reconstruct the input

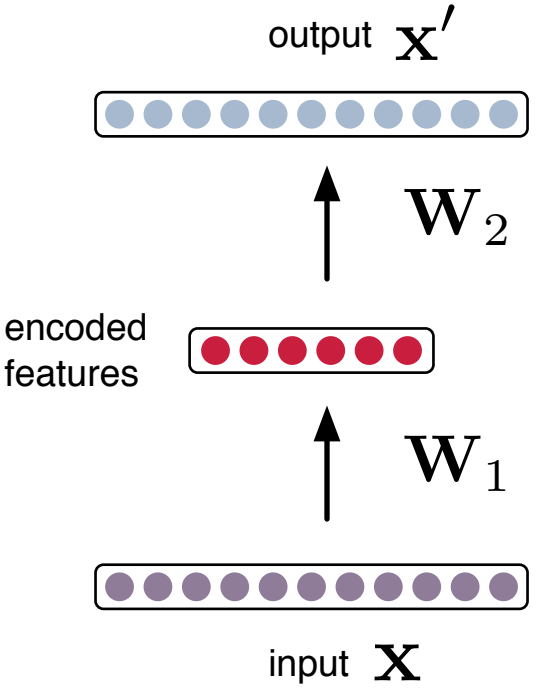
$$\mathbf{x}' = \mathbf{W}_2^{\mathsf{T}}\mathbf{z}$$

- Objective function: Minimizing reconstruction loss

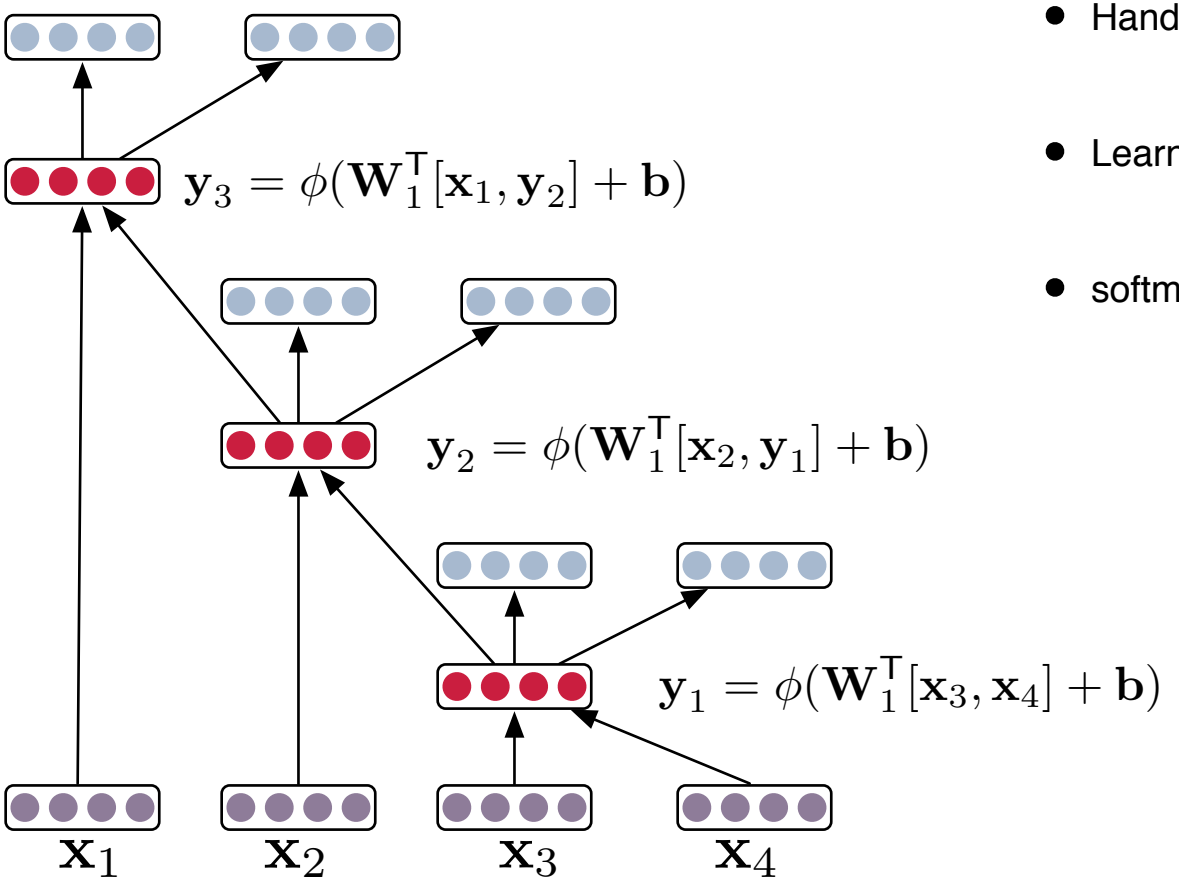
$$\min ||\mathbf{x}' - \mathbf{x}||_2$$

- Use in unsupervised/semi-supervised settings

- Learn invariant features from the inputs



Recursive Auto Encoder



- Handle variable input lengths
- Learn multilevel of representation
- softmax at every node of the tree

Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions.
Richard Socher, Jeffrey Pennington, Eric Huang, Andrew Y. Ng, and Christopher D. Manning.
EMNLP 2011

Objectives

- Maximizing log-likelihood (cross Entropy criterion)

$$\sum_{k \in D} \ln p(y_k | x_k)$$

$$\sum_{k \in D} \sum_{i=1}^{n} t_i \ln p(y_k^i | x_k)$$

$$t_i \in \{0, 1\}, \sum_i^n t_i = 1$$

- Max-margin criterion
  - Non probabilistic approach
  - Generate negative examples
  - Discriminate bad examples from good one

$$\sum_{i=1}^{k} \max(0, \mathbf{y}_{good}\mathbf{h} - \mathbf{y}_{bad}^i\mathbf{h} - \underbrace{m}_{\text{margin}})$$

- Many other objective functions

https://github.com/torch/nn/blob/master/doc/criterion.md#nn.Criterions

Tunable Metric

- Query representations and target representations have different dimension

Learn transformation matrices

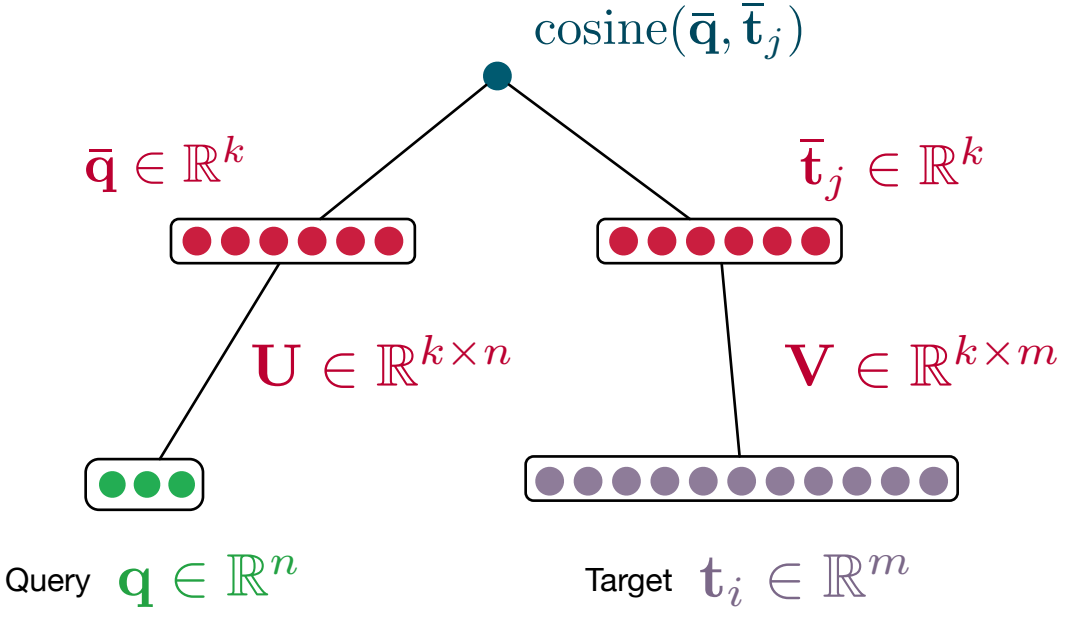
$$\mathbf{U} \in \mathbb{R}^{k \times n}$$

$$\mathbf{V} \in \mathbb{R}^{k \times m}$$

- Similarity function is differentiable

$$\frac{\partial}{\partial \mathbf{U}} \text{cosine}(\bar{\mathbf{q}}, \bar{\mathbf{t}}_j)$$

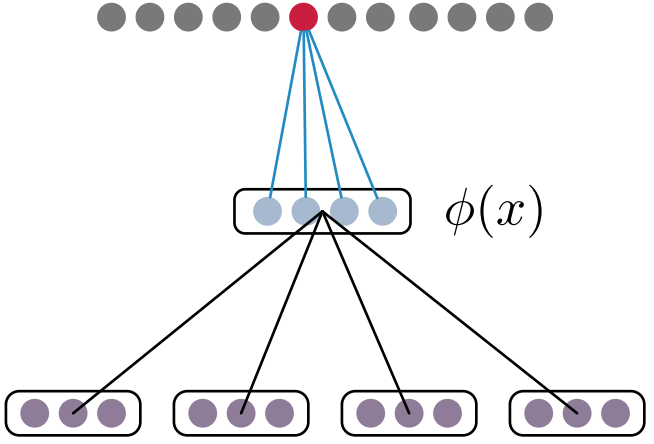
$$\frac{\partial}{\partial \mathbf{V}} \text{cosine}(\bar{\mathbf{q}}, \bar{\mathbf{t}}_j)$$



- Typically train with max-margin like objectives

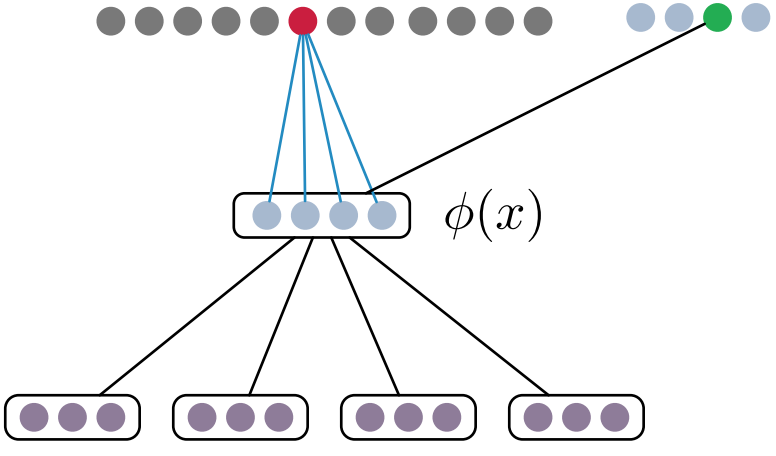
Handling large number of outputs

- Training language model with (feed-forward, recurrent) NNs, number of outputs can be up to few hundreds of thousands

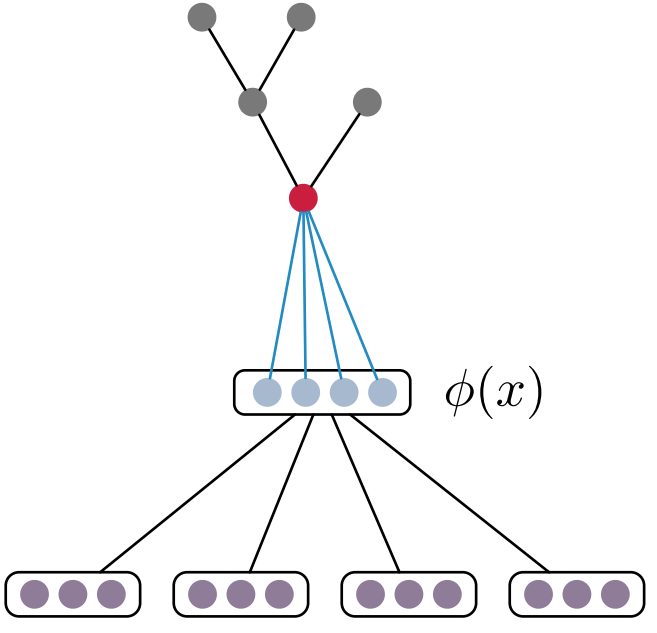


$$(n-1)d \times H + \boxed{H \times |V|}$$

complexity



$$p(w_j|\text{hist}) = p(C_i|\text{hist})p(w_j|C_i)$$



Hierarchical softmax

Take home message

- NNs are a special case of graphical model

  Sum product/Max sum algorithm

- From words to sentences/paragraphs/documents

  by mean of compositionally

- Combine different type of NNs

  Convolution NN with Recurrent NN

- Probabilistic vs. non probabilistic objectives

- Define your own models