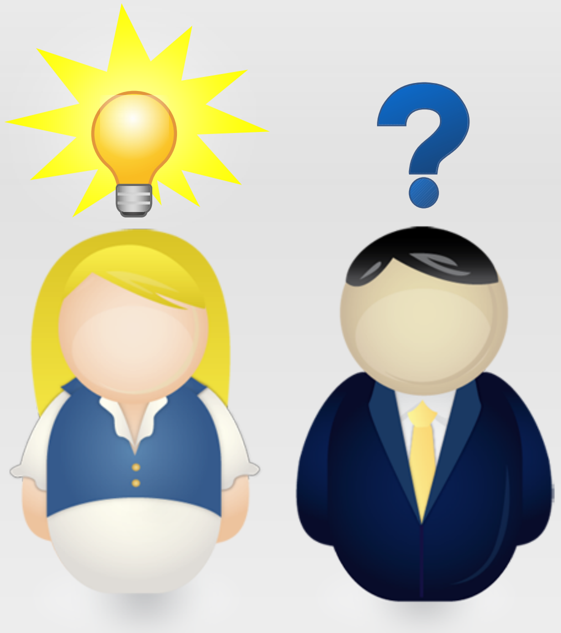
Rudolf Rosa
rosa@ufal.mff.cuni.cz

Deep Neural Networks in Natural Language Processing

Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

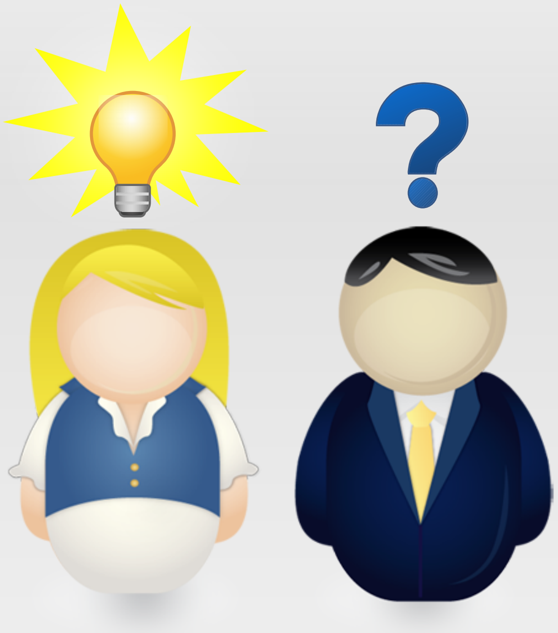
Hora Informaticae, ÚI AV ČR, Praha, 14 Jan 2019

Background check: do you know...



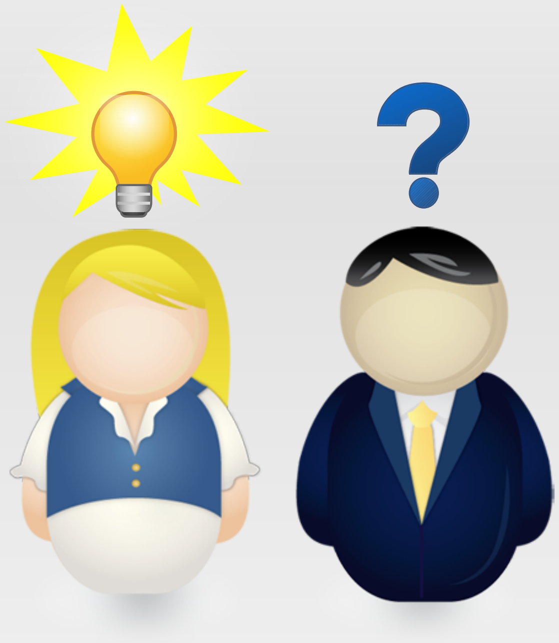
Background check: do you know...

- Machine learning? (ML)



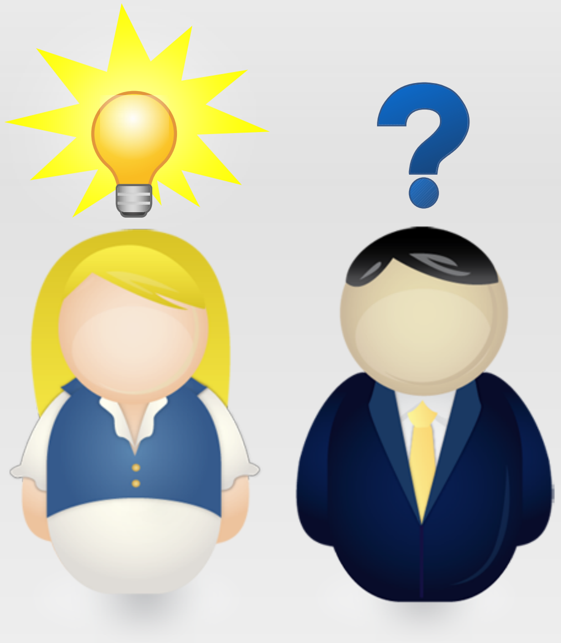
Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)



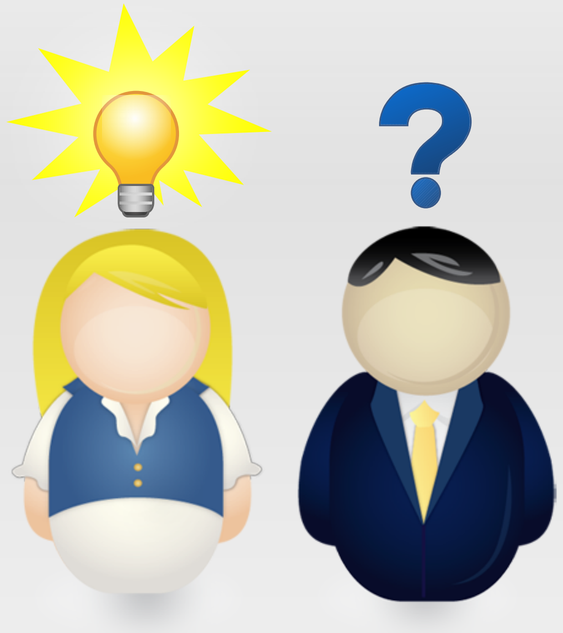
Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)



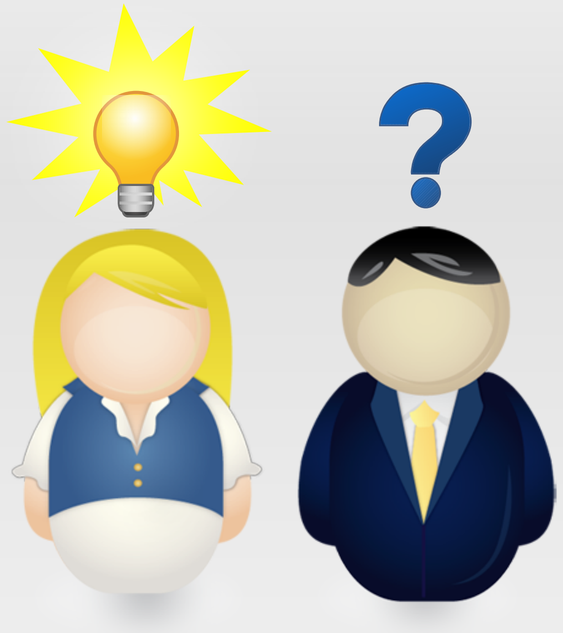
Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)



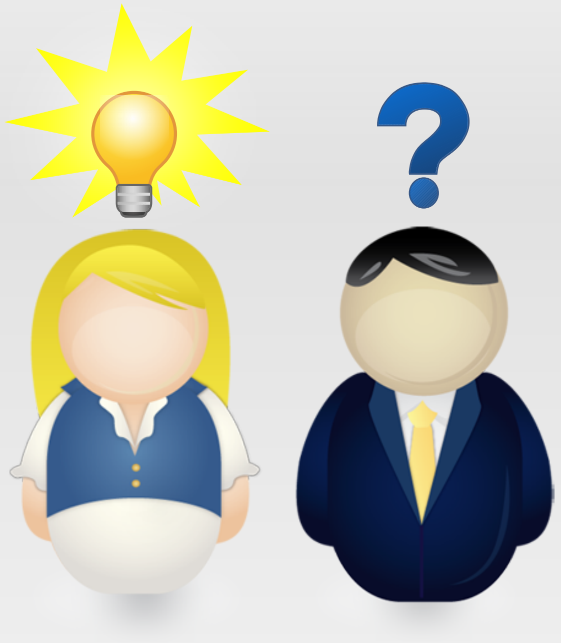
Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)
- Recurrent neural networks? (RNN)



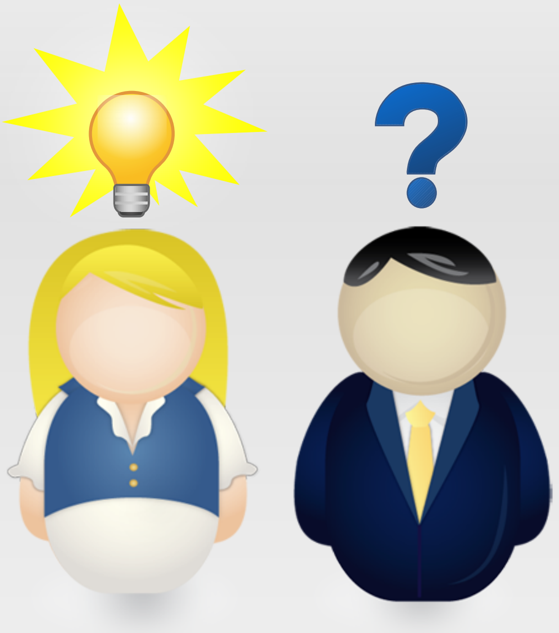
Background check: do you know...

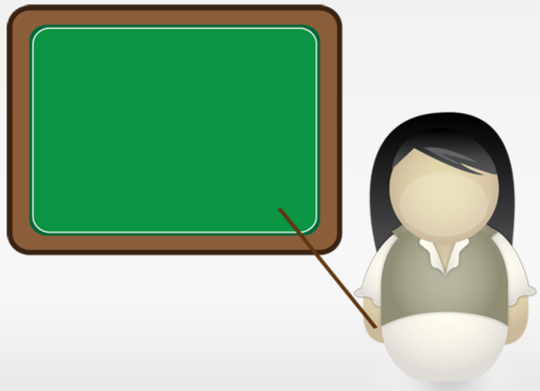
- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)
- Recurrent neural networks? (RNN)
  - Long short-term memory units? (LSTM)
  - Gated recurrent units? (GRU)



Background check: do you know...

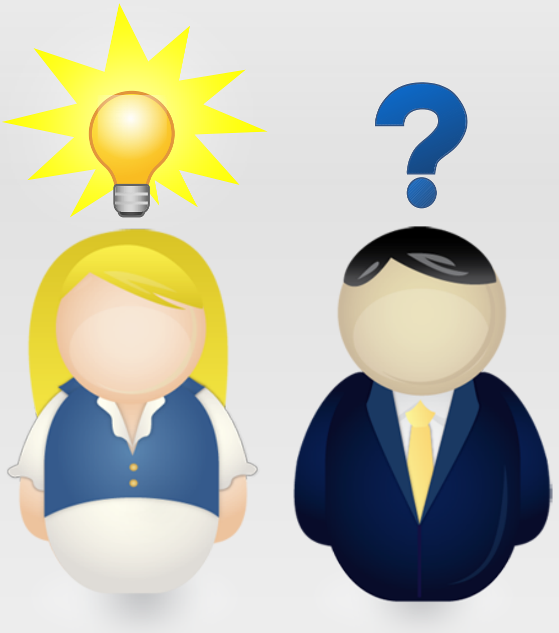
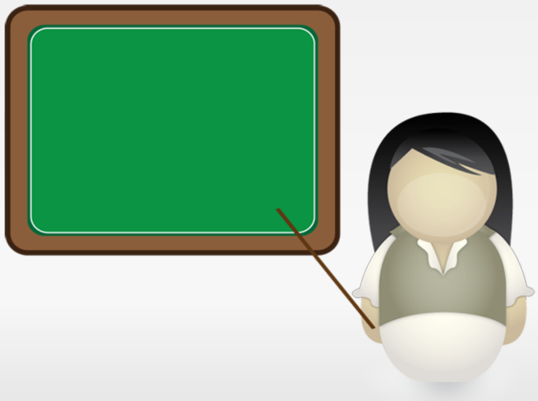
- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)
- Recurrent neural networks? (RNN)
  - Long short-term memory units? (LSTM)
  - Gated recurrent units? (GRU)
- Attention mechanism? (Bahdanau+, 2014)





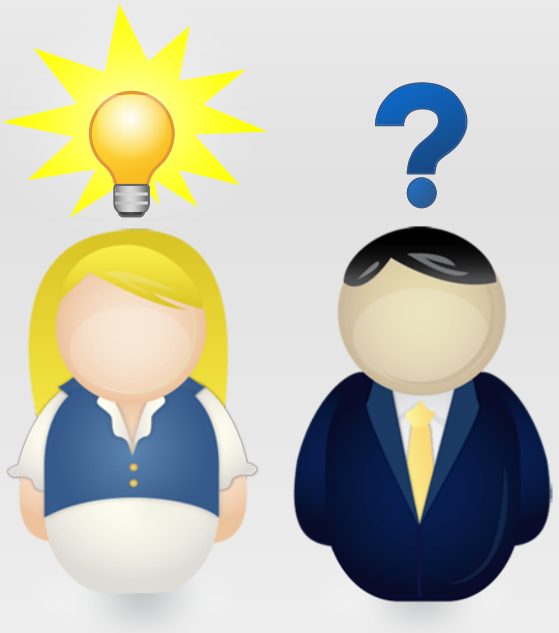
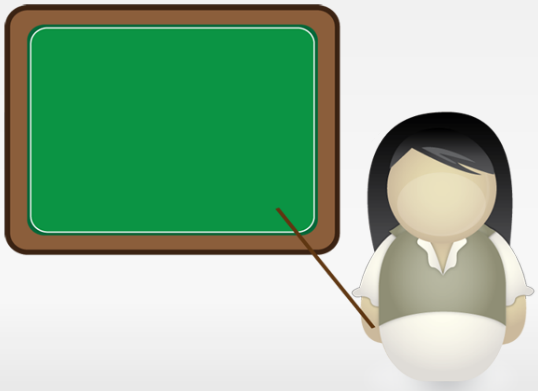
Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)
- Recurrent neural networks? (RNN)
    - Long short-term memory units? (LSTM)
    - Gated recurrent units? (GRU)
- Attention mechanism? (Bahdanau+, 2014)
    - Self-attentive networks? (SAN, Transformer)

Background check: do you know...

- Machine learning? (ML)
- Artificial neural networks? (NN)
- Deep neural networks? (DNN)
- Convolutional neural networks? (CNN)
- Recurrent neural networks? (RNN)
  - Long short-term memory units? (LSTM)
  - Gated recurrent units? (GRU)
- Attention mechanism? (Bahdanau+, 2014)
  - Self-attentive networks? (SAN, Transformer)
- Word embeddings? (Bengio+, 2003)
  - Word2vec? (Mikolov+, 2013)

ML in Natual Language Processing

- Before: complex multistep pipelines
  - Preprocessing, low-level processing, high-level processing, classification, post-processing…
  - Massive feature engineering, linguistic knowledge…

ML in Natual Language Processing

- Before: complex multistep pipelines
  - Preprocessing, low-level processing, high-level processing, classification, post-processing…
  - Massive feature engineering, linguistic knowledge…
- Now: monolitic end-to-end systems (or nearly)

ML in Natual Language Processing

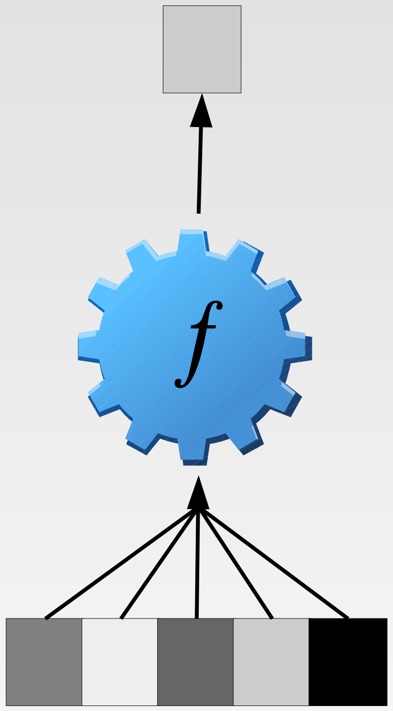
- Before: complex multistep pipelines
  - Preprocessing, low-level processing, high-level processing, classification, post-processing…
  - Massive feature engineering, linguistic knowledge…
- Now: monolitic end-to-end systems (or nearly)
  - text → deep neural network → output
  - Little or no linguistic knowledge required
  - Little or no feature engineering
  - Little or no dependence on external tools

ML in Natual Language Processing

- Before: complex multistep pipelines
  - Preprocessing, low-level processing, high-level processing, classification, post-processing…
  - Massive feature engineering, linguistic knowledge…
- Now: monolitic end-to-end systems (or nearly)
  - text → deep neural network → output
  - Little or no linguistic knowledge required
  - Little or no feature engineering
  - Little or no dependence on external tools
  - → so now is a good time for **anyone** to get into NLP!

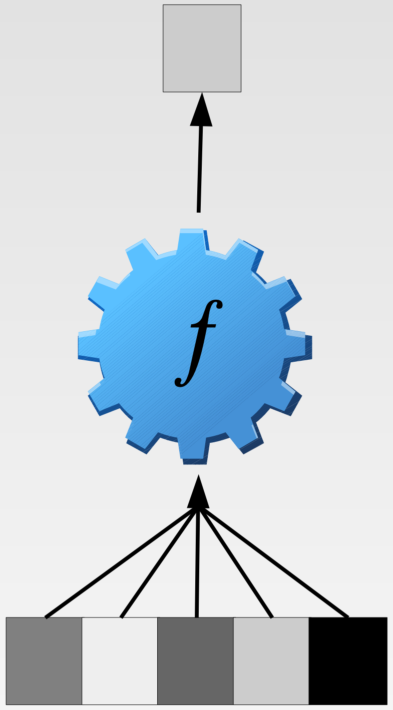
Neural networks & text processing

- Input to a neuron: <u>fixed-dimension</u> <u>real</u> vector



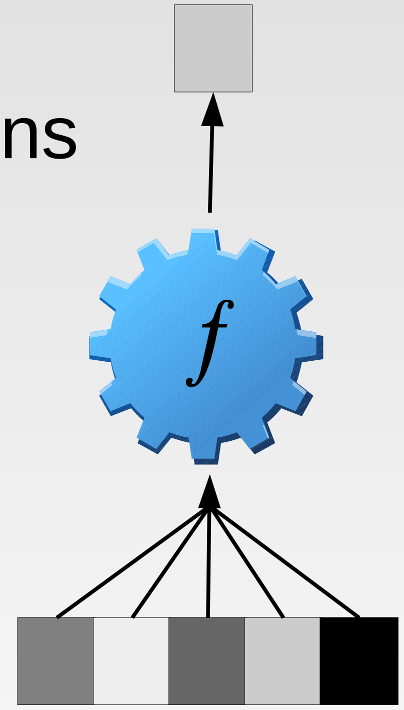
Neural networks & text processing

- Input to a neuron: <u>fixed-dimension</u> <u>real</u> vector
  - Dimension should be reasonable (<10^3)



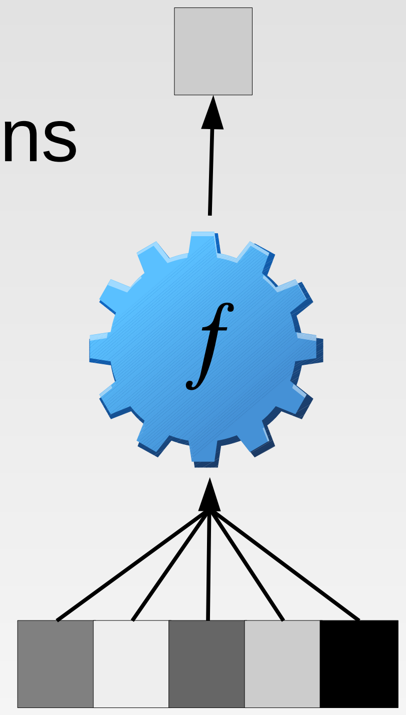
Neural networks & text processing

- Input to a neuron: <u>fixed-dimension</u> <u>real</u> vector
  - Dimension should be reasonable ($<10^3$)
  - Neural net: fixed-sized network of neurons



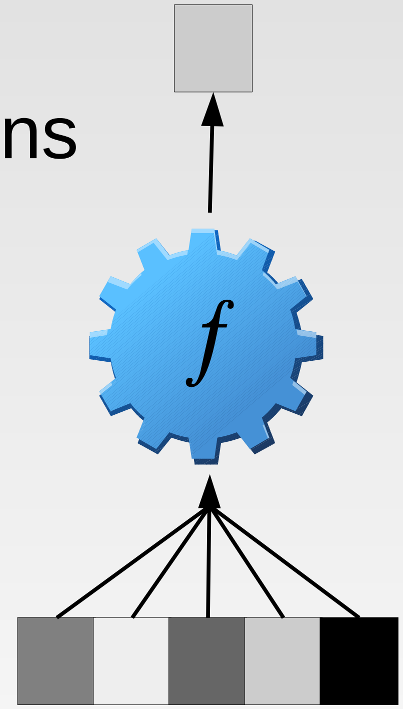
Neural networks & text processing

- Input to a neuron: fixed-dimension real vector
  - Dimension should be reasonable (<10^3)
  - Neural net: fixed-sized network of neurons
- Text input: sequence processing
  - Sentence = sequence of words



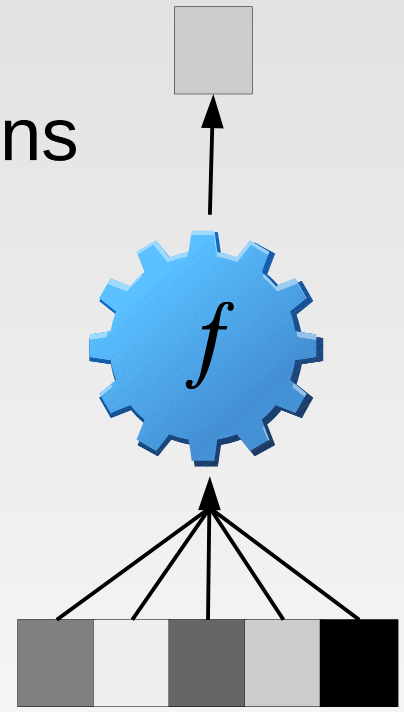
Neural networks & text processing

- Input to a neuron: fixed-dimension real vector
  - Dimension should be reasonable ($<10^3$)
  - Neural net: fixed-sized network of neurons
- Text input: sequence processing
  - Sentence = sequence of words
  - Words: discrete (but interrelated)
    - Massively multi-valued ($\sim 10^6$)
    - Very sparse (Zipf distribution)



Neural networks & text processing

- Input to a neuron: <u>fixed-dimension</u> <u>real</u> vector
  - Dimension should be reasonable (<10^3)
  - Neural net: fixed-sized network of neurons
- Text input: sequence processing
  - Sentence = sequence of words
  - Words: discrete (but interrelated)
    - Massively multi-valued (~10^6)
    - Very sparse (Zipf distribution)
  - Sentences: variable length (~1 to 100)
    - Complex and hidden internal structure



Outline of the talk

- Problem 1: Words
- Problem 2: Sentences

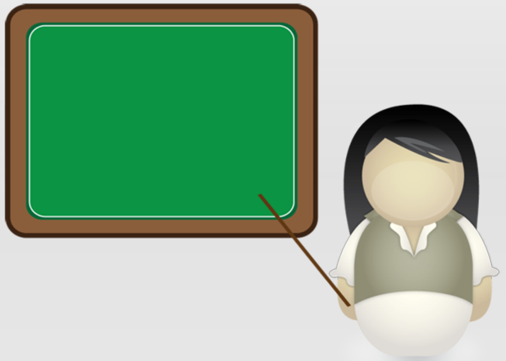
Outline of the talk

- Problem 1: Words
    - There are too many
    - They are discrete
    - *Representing massively multi-valued discrete data by continuous low-dimensional vectors*
- Problem 2: Sentences

Outline of the talk

- Problem 1: Words
  - There are too many
  - They are discrete
  - *Representing massively multi-valued discrete data by continuous low-dimensional vectors*
- Problem 2: Sentences
  - They have various lengths
  - They have internal structure
  - *Handling variable-length input sequences with complex internal relations by fixed-sized neural units*

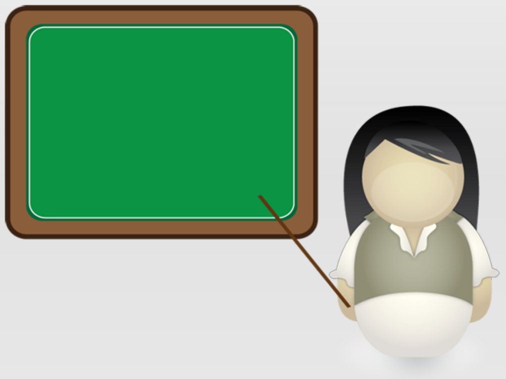
Outline of the talk



- Problem 1: Words
  - There are too many
  - They are discrete
  - *Representing massively multi-valued discrete data by continuous low-dimensional vectors*
- Problem 2: Sentences
  - They have various lengths
  - They have internal structure
  - *Handling variable-length input sequences with complex internal relations by fixed-sized neural units*

Outline of the talk

- Problem 1: Words
  - There are too many
  - They are discrete
  - *Representing massively multi-valued discrete data by continuous low-dimensional vectors*



- Problem 2: Sentences
  - They have various lengths
  - They have internal structure
  - *Handling variable-length input sequences with complex internal relations by fixed-sized neural units*

Warnings

Warnings

- I am not a ML expert, rather a ML *user*
  - Please excuse any errors and inaccuracies

Warnings

- I am not a ML expert, rather a ML *user*
  - Please excuse any errors and inaccuracies
- Focus of talk: input representation (“encoding”)
  - Key problem in NLP, interesting properties

Warnings

- I am not a ML expert, rather a ML *user*
  - Please excuse any errors and inaccuracies
- Focus of talk: input representation (“encoding”)
  - Key problem in NLP, interesting properties
- Leaving out
  - Generating output (“decoding”) – that’s also interesting

Warnings ⚠

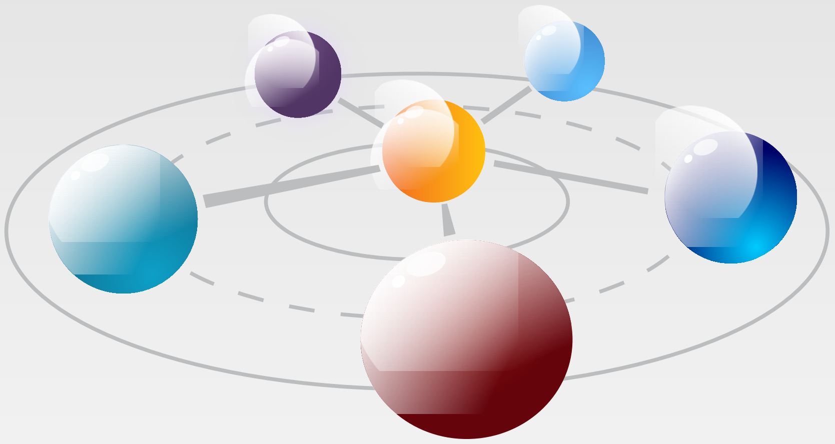
- I am not a ML expert, rather a ML *user*
  - Please excuse any errors and inaccuracies
- Focus of talk: input representation (“encoding”)
  - Key problem in NLP, interesting properties
- Leaving out
  - Generating output (“decoding”) – that’s also interesting
    - Sequence generation
      - Seq. elements discrete, large domain (softmax over 10^6)
      - Sequence length not a priori known

Warnings

- I am not a ML expert, rather a ML *user*
  - Please excuse any errors and inaccuracies
- Focus of talk: input representation (“encoding”)
  - Key problem in NLP, interesting properties
- Leaving out
  - Generating output (“decoding”) – that’s also interesting
    - Sequence generation
      - Seq. elements discrete, large domain (softmax over 10^6)
      - Sequence length not a priori known
  - Decision at encoder/decoder boundary (if any)

Problem 1: Words

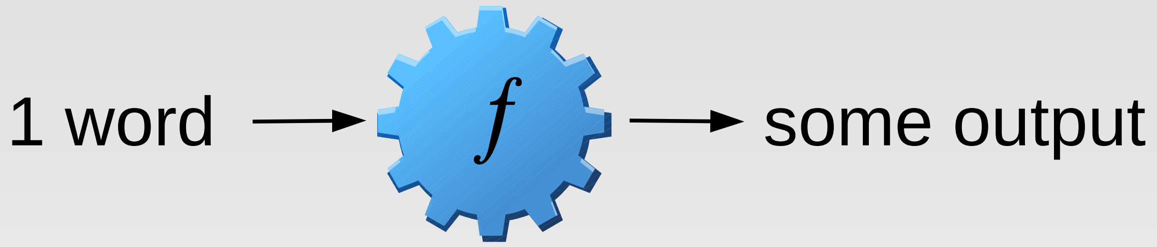
Massively multi-valued discrete data
(words)



Continuous low-dimensional vectors
(word embeddings)

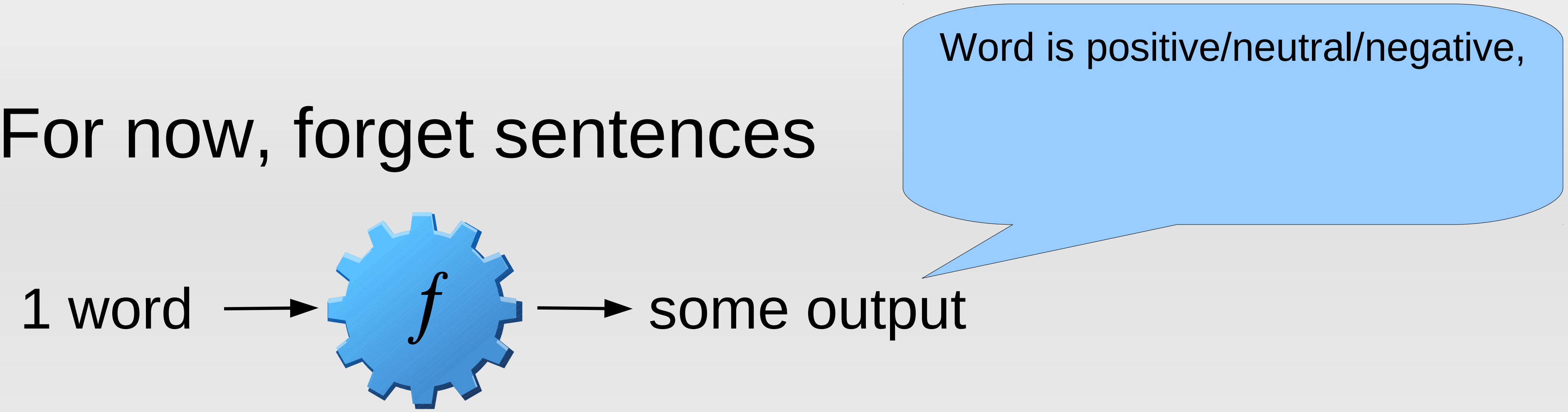
Simplification

- For now, forget sentences



Simplification

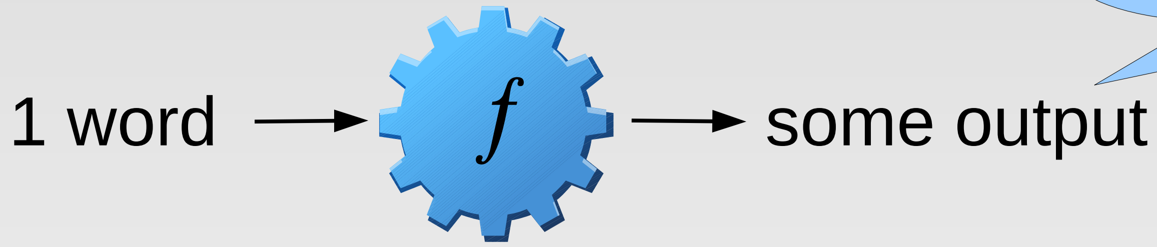
- For now, forget sentences



Simplification

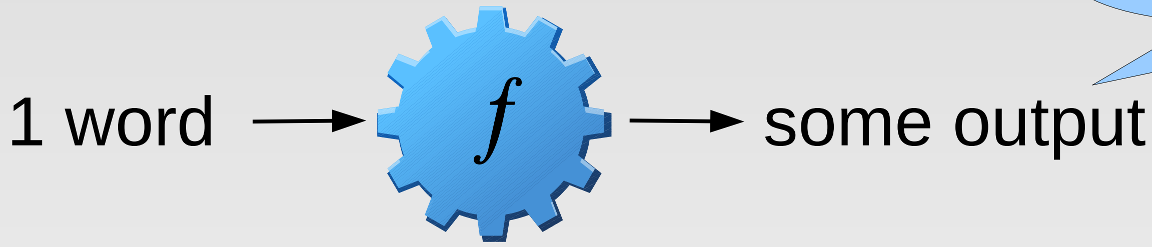
Word is positive/neutral/negative,
Definition of the word,

- For now, forget sentences



Simplification

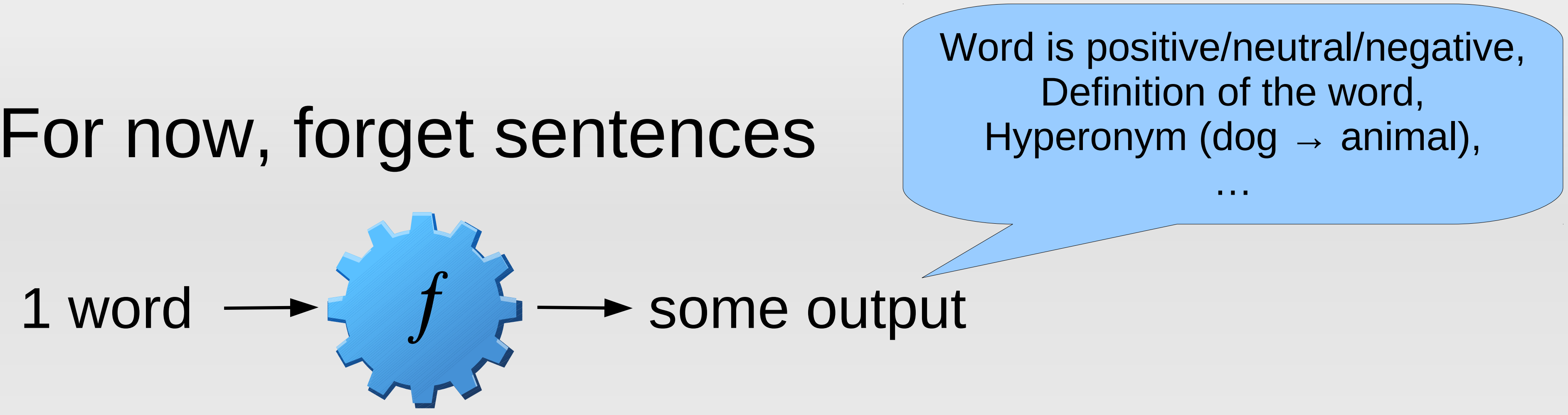
- For now, forget sentences



Word is positive/neutral/negative,
Definition of the word,
Hyperonym (dog → animal),
...

Simplification

- For now, forget sentences



- Situation
  - We have labelled training data for **some** words (10^3)
  - We want to generalize (ideally) to **all** words (10^6)

The problem with words

- How many words are there?

The problem with words

- How many words are there? Too many!

The problem with words

- How many words are there? Too many!
  - Many problems with counting words, cannot be done

The problem with words

- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6

The problem with words

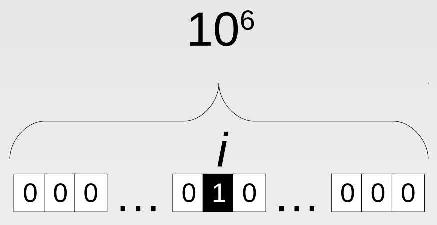
- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)

The problem with words

- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)
- Long-standing problem of NLP

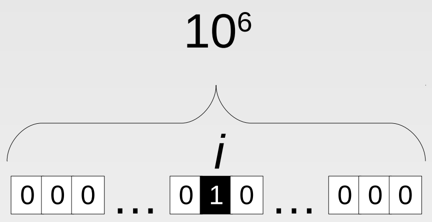
The problem with words

- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)
- Long-standing problem of NLP
- Natural representation: 1-hot vector



The problem with words

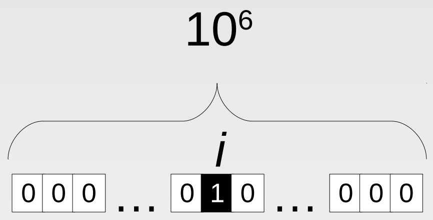
- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)
- Long-standing problem of NLP
- Natural representation: 1-hot vector



  - ML with ~10^6 binary features on input

The problem with words

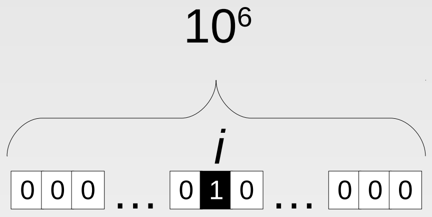
- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)
- Long-standing problem of NLP
- Natural representation: 1-hot vector


$$\overbrace{\boxed{0\,0\,0}\ \dots\ \boxed{0\,\overset{i}{1}\,0}\ \dots\ \boxed{0\,0\,0}}^{10^6}$$

  - ML with ~10^6 binary features on input 😒
  - Pair of words: ~10^{12} 😠

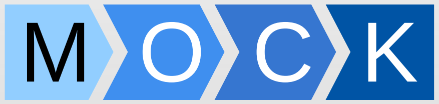
The problem with words

- How many words are there? Too many!
  - Many problems with counting words, cannot be done
  - ~10^6 (but potentially infinite – new words get created every day)
- Long-standing problem of NLP
- Natural representation: 1-hot vector



  - ML with ~10^6 binary features on input
  - Pair of words: ~10^{12}
  - No generalization, meaning of words not captured
    - dog~puppy, dog~~cat, dog~~~platypus, dog~~~~whiskey

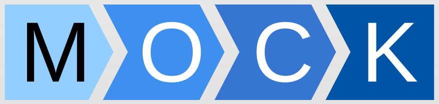
Split the words

- Split into characters 

Split the words

- Split into characters M O C K
  - Not that many (~10^2)

Split the words

- Split into characters 

  - Not that many (~10^2)
  - Do not capture meaning
    - Starts with “m-”, is it positive or negative?

Split the words

- Split into characters M O C K
  - Not that many (~10^2)
  - Do not capture meaning
    - Starts with “m-”, is it positive or negative?
- Split into subwords/morphemes mis class if ied

Split the words

- Split into characters M O C K
  - Not that many (~10^2) 😊
  - Do not capture meaning 😒
    - Starts with “m-”, is it positive or negative?
- Split into subwords/morphemes mis class if ied
  - Word starts with “mis-”: it is probably negative
    - *misclassify, mistake, misconception…*

Split the words

- Split into characters M O C K
  - Not that many (~10^2)
  - Do not capture meaning
    - Starts with “m-”, is it positive or negative?
- Split into subwords/morphemes mis class if ied
  - Word starts with “mis-”: it is probably negative
    - *misclassify, mistake, misconception…*
  - Helps, used in practice

Split the words

- Split into characters M O C K
  - Not that many (~10^2)
  - Do not capture meaning
    - Starts with “m-”, is it positive or negative?
- Split into subwords/morphemes mis class if ied
  - Word starts with “mis-”: it is probably negative
    - *misclassify, mistake, misconception…*
  - Helps, used in practice
    - Potentially infinite set covered by a finite set of subwords

Split the words

- Split into characters M O C K
  - Not that many ($\sim 10^2$) 😊
  - Do not capture meaning 😒
    - Starts with “m-”, is it positive or negative?
- Split into subwords/morphemes mis class if ied
  - Word starts with “mis-”: it is probably negative
    - *misclassify, mistake, misconception…*
  - Helps, used in practice 😊
    - Potentially infinite set covered by a finite set of subwords
  - Meaning-capturing subwords still too many ($\sim 10^5$) 😒

Distributional hypothesis

Distributional hypothesis

- *smelt* (assume you don’t know this word)

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.*

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.*

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → animal/illness

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → <u>animal</u>/illness

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → animal/illness
  - ***Smelts** are disappearing from oceans.*

Distributional hypothesis

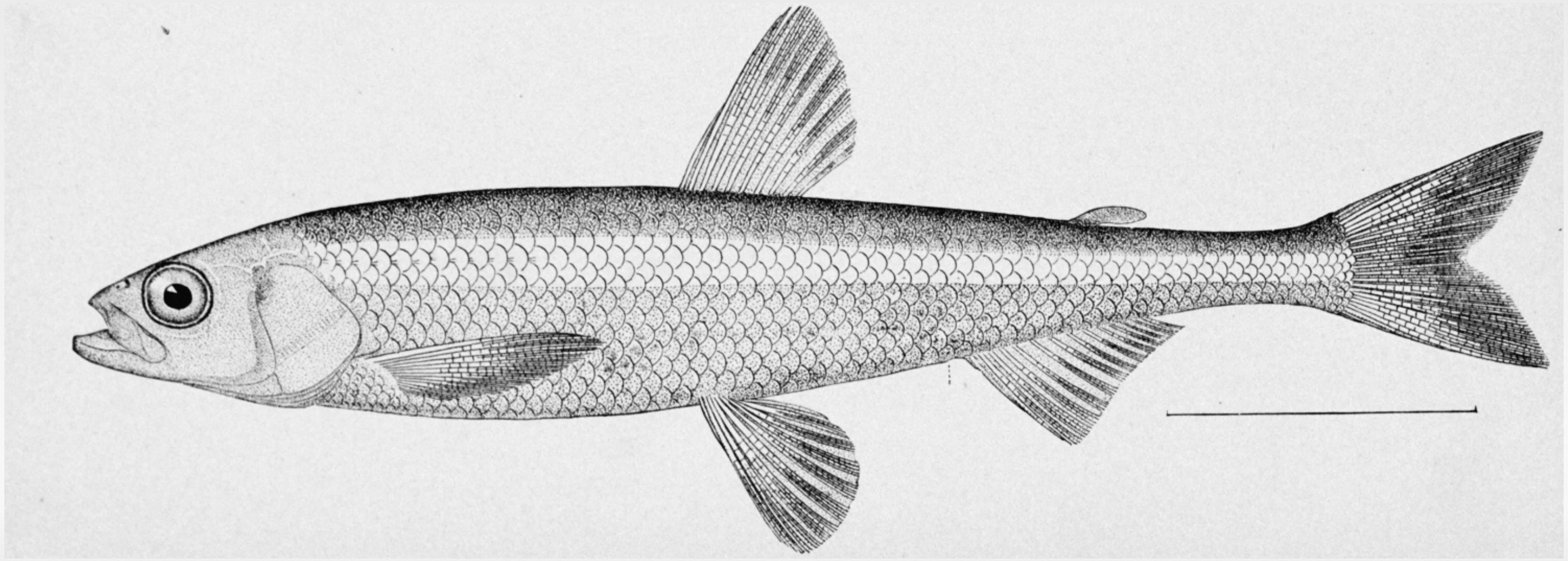
- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → <u>animal</u>/illness
  - ***Smelts** are disappearing from oceans.* → plant/fish

Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → <u>animal</u>/illness
  - ***Smelts** are disappearing from oceans.* → plant/<u>fish</u>

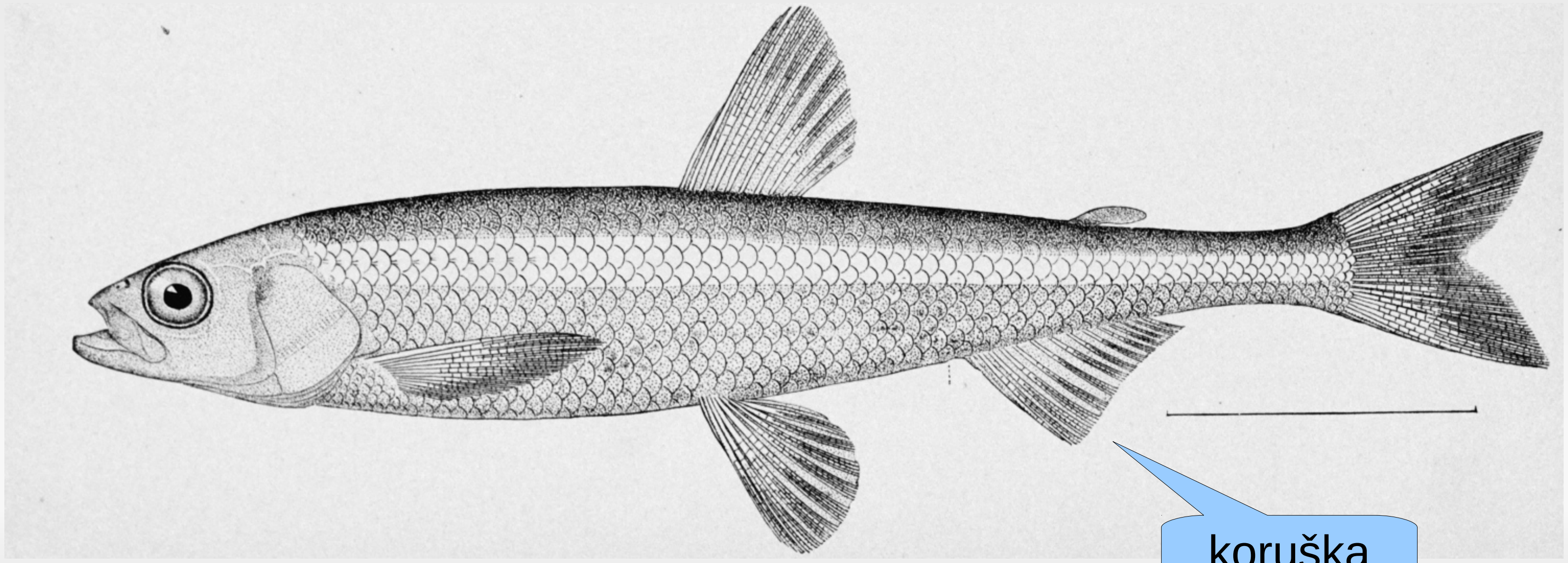
Distributional hypothesis

- *smelt* (assume you don't know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → animal/illness
  - ***Smelts** are disappearing from oceans.* → plant/fish



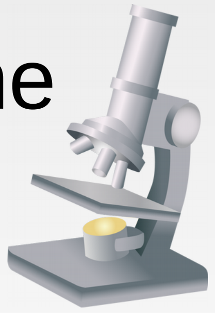
Distributional hypothesis

- *smelt* (assume you don’t know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → animal/illness
  - ***Smelts** are disappearing from oceans.* → plant/fish



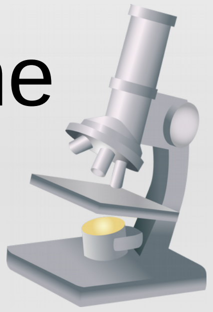
Distributional hypothesis

- *smelt* (assume you don't know this word)
  - *I had a **smelt** for lunch.* → noun, meal/food
  - *My father caught a **smelt**.* → <u>animal</u>/illness
  - ***Smelts** are disappearing from oceans.* → plant/<u>fish</u>
- Harris (1954): "Words that occur in the same contexts tend to have similar meanings."



Distributional hypothesis

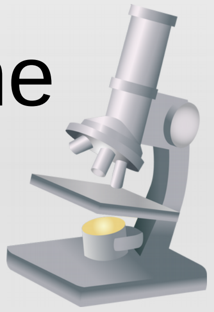
- Harris (1954): “Words that occur in the same contexts tend to have similar meanings.”
- Cooccurrence matrix
  - # of sentences containing both WORD and CONTEXT



WORD	CONTEXT				
	lunch	caught	oceans	doctor	green
smelt	10	10	10	1	1
salmon	100	100	100	1	1
flu	1	100	1	100	10
seaweed	10	1	100	1	100

Distributional hypothesis

- Harris (1954): “Words that occur in the same contexts tend to have similar meanings.”
- Cooccurrence matrix
  - # of sentences containing both WORD and CONTEXT

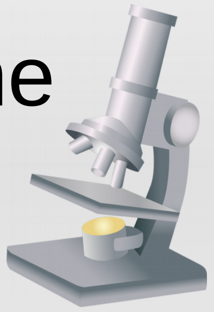


WORD	CONTEXT				
	lunch	caught	oceans	doctor	green
smelt	10	10	10	1	1
salmon	100	100	100	1	1
flu	1	100	1	100	10
seaweed	10	1	100	1	100

  - Cheap plentiful data (webs, news, books…): ~10^9

Distributional hypothesis

- Harris (1954): “Words that occur in the same contexts tend to have similar meanings.”
- Cooccurrence matrix
  - # of sentences containing both WORD and CONTEXT



WORD	CONTEXT				
	lunch	caught	oceans	doctor	green
smelt	10	10	10	1	1
salmon	100	100	100	1	1
flu	1	100	1	100	10
seaweed	10	1	100	1	100

NxN, $N \sim 10^6$

  - Cheap plentiful data (webs, news, books…): $\sim 10^9$

From cooccurence to PMI

Association measures

- Cooccurrence matrix
  - $M_C[i, j] = \text{count}(\text{word}_i \,\&\, \text{context}_j)$
- Conditional probability matrix
  - $M_P[i, j] = P(\text{word}_i \mid \text{context}_j) = M_C[i, j] / \text{count}(\text{context}_j)$
- Conditional log-probability matrix
  - $M_{LogP}[i, j] = \log P(\text{word}_i \mid \text{context}_j) = \log M_P[i, j]$
- Pointwise mutual information matrix
  - $M_{PMI}[i, j] = \log [P(\text{word}_i \mid \text{context}_j) / P(\text{word}_i)]$

From cooccurence to PMI

Association measures

- Cooccurrence matrix
  - $M_C[i, j] = \text{count}(\text{word}_i \,\&\, \text{context}_j)$
- Conditional probability matrix
  - $M_P[i, j] = P(\text{word}_i \mid \text{context}_j) = M_C[i, j] / \text{count}(\text{context}_j)$
- Conditional log-probability matrix
  - $M_{LogP}[i, j] = \log P(\text{word}_i \mid \text{context}_j) = \log M_P[i, j]$
- Pointwise mutual information matrix
  - $M_{PMI}[i, j] = \log [P(\text{word}_i \mid \text{context}_j) / P(\text{word}_i)]$
  - $PMI(A, B) = \log P(A \,\&\, B) / P(A)\, P(B)$

From cooccurence to PMI

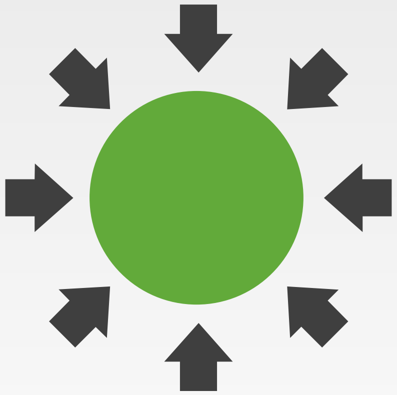
- Word representation still impratically huge 😟
  - $M_{PMI}[i] \in \mathbf{R}^N$, $N \sim 10^6$

From cooccurence to PMI

- Word representation still impratically huge 😟
  - $M_{PMI}[i] \in \mathbf{R}^N$, $N \sim 10^6$
- But better than 1-hot 😊
  - Meaningful continuous vectors (e.g. cos similarity)

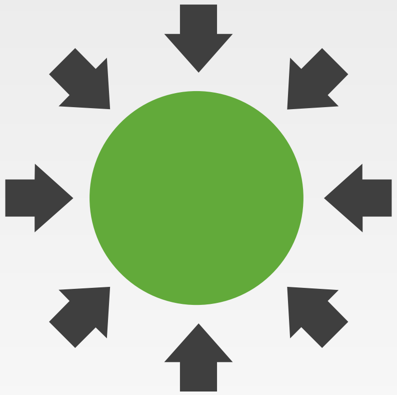
From cooccurence to PMI

- Word representation still impratically huge
  - $M_{PMI}[i] \in \mathbf{R}^N$, $N \sim 10^6$
- But better than 1-hot
  - Meaningful continuous vectors (e.g. cos similarity)
- Just need to compress it!



From cooccurence to PMI

- Word representation still impratically huge 😒
  - $M_{PMI}[i] \in \mathbf{R}^N$, $N \sim 10^6$
- But better than 1-hot 😊
  - Meaningful continuous vectors (e.g. cos similarity)
- Just need to compress it!
  - Explicitly: matrix factorization
    - post-hoc, not used
  - Implicitly: word2vec
    - widely used



Matrix factorization

Matrix factorization

- Levy&Goldberg (2014)

Matrix factorization

- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}

Matrix factorization

- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}
- Shift the matrix to make it positive (- min)

Matrix factorization

- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}
- Shift the matrix to make it positive (- min)
- Truncated Singular Value Decomposition:
  - $\overline{M} = UDV^T$ $\quad M \in \mathbf{R}^{N\text{x}N} \rightarrow U \in \mathbf{R}^{N\text{x}d}, D \in \mathbf{R}^{d\text{x}d}, V \in \mathbf{R}^{N\text{x}d}$

$N \sim 10^6$
$d \sim 10^2$

Matrix factorization

- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}
- Shift the matrix to make it positive (- min)
- Truncated Singular Value Decomposition:
  - $\overline{M} = UDV^T$ $M \in \mathbf{R}^{NxN} \rightarrow U \in \mathbf{R}^{Nxd}, D \in \mathbf{R}^{dxd}, V \in \mathbf{R}^{Nxd}$
- Word embedding matrix: $W = UD \in \mathbf{R}^{Nxd}$
  - Embedding $vec(word_i) = W[i] \in \mathbf{R}^d$

$N \sim 10^6$
$d \sim 10^2$

Matrix factorization

- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}
- Shift the matrix to make it positive (- min)
- Truncated Singular Value Decomposition:
  - $\overline{M} = UDV^T \qquad M \in \mathbf{R}^{NxN} \rightarrow U \in \mathbf{R}^{Nxd}, D \in \mathbf{R}^{dxd}, V \in \mathbf{R}^{Nxd}$
- Word embedding matrix: $W = UD \in \mathbf{R}^{Nxd}$
  - Embedding $vec(word_i) = W[i] \in \mathbf{R}^d$
  - Continuous low-dimensional vector 😊

$N \sim 10^6$
$d \sim 10^2$

Matrix factorization

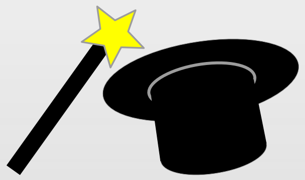
- Levy&Goldberg (2014)
- Take M_{LogP} or M_{PMI}
- Shift the matrix to make it positive (- min)
- Truncated Singular Value Decomposition:
  - $\overline{M} = UDV^T$ $M \in \mathbf{R}^{NxN} \rightarrow U \in \mathbf{R}^{Nxd}, D \in \mathbf{R}^{dxd}, V \in \mathbf{R}^{Nxd}$
- Word embedding matrix: $W = UD \in \mathbf{R}^{Nxd}$
  - Embedding $vec(word_i) = W[i] \in \mathbf{R}^d$
  - Continuous low-dimensional vector 😊
  - Meaningful (cos similarity, algebraic operations) 😊

$N \sim 10^6$
$d \sim 10^2$

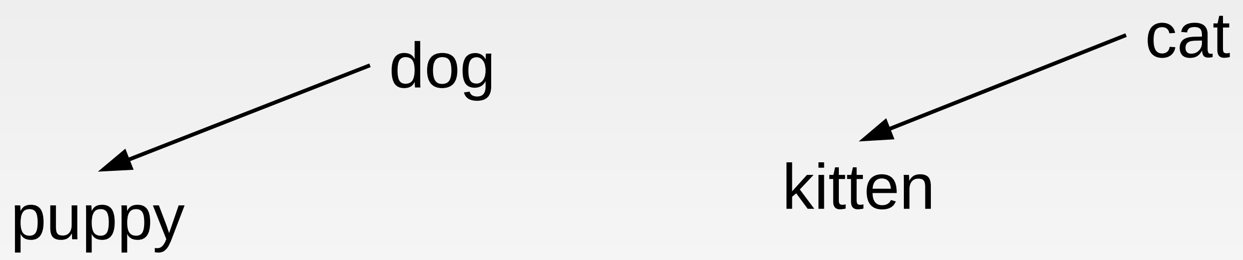
Word embeddings magic

- Word similarity (cos)
  - vec(dog) ~ vec(puppy),   vec(cat) ~ vec(kitten)

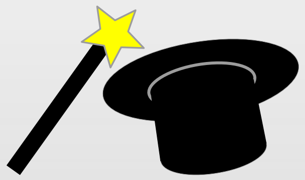
Word embeddings magic



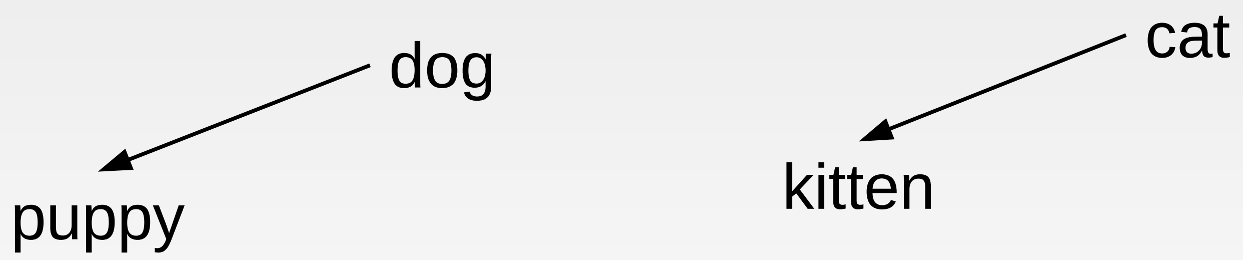
- Word similarity (cos)
  - vec(dog) ~ vec(puppy),   vec(cat) ~ vec(kitten)
- Word meaning algebra
  - Some relations parallel across words
  - vec(puppy) - vec(dog)   ~   vec(kitten) - vec(cat)



Word embeddings magic

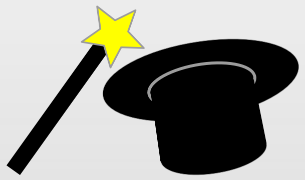


- Word similarity (cos)
  - vec(dog) ~ vec(puppy), vec(cat) ~ vec(kitten)
- Word meaning algebra
  - Some relations parallel across words
  - vec(puppy) - vec(dog) ~ vec(kitten) - vec(cat)

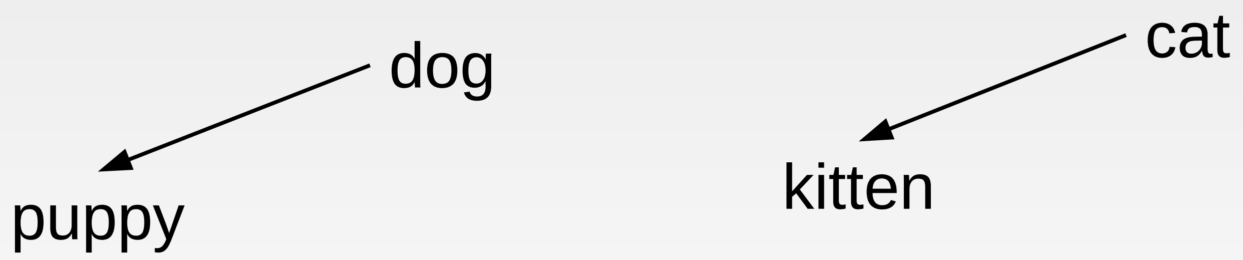


  - => vec(puppy) - vec(dog) + vec(cat) ~ vec(kitten)

Word embeddings magic



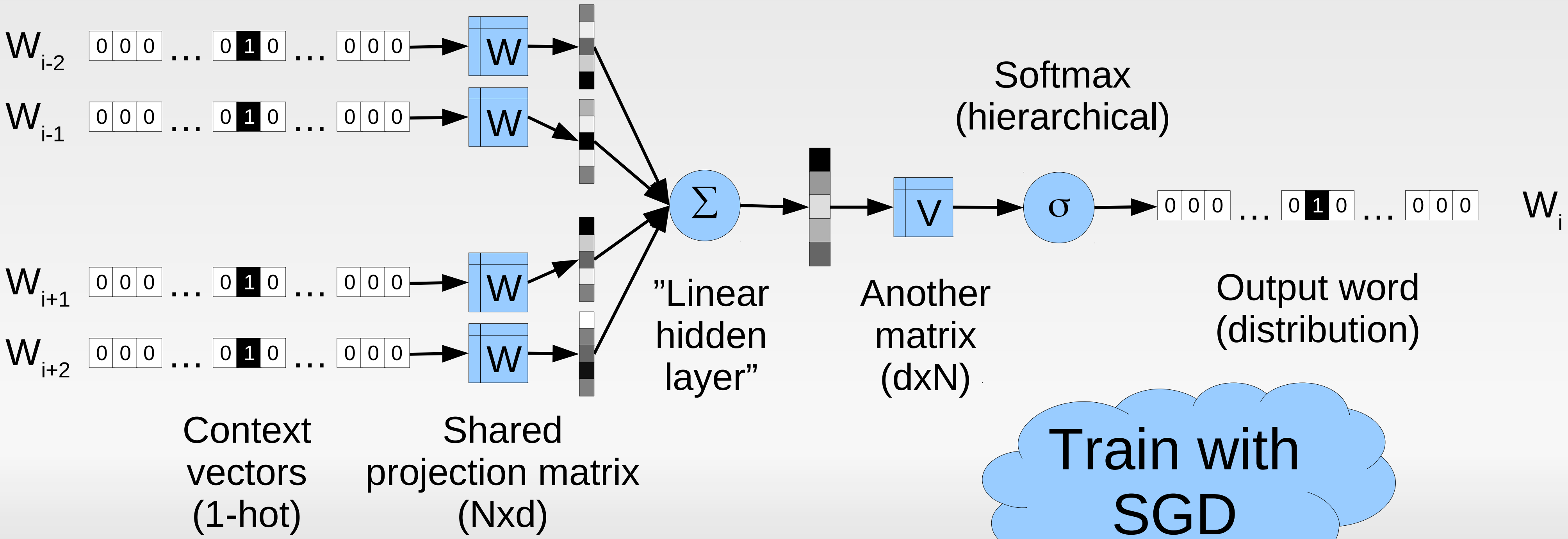
- Word similarity (cos)
  - vec(dog) ~ vec(puppy), vec(cat) ~ vec(kitten)
- Word meaning algebra
  - Some relations parallel across words
  - vec(puppy) - vec(dog) ~ vec(kitten) - vec(cat)



  - => vec(puppy) - vec(dog) + vec(cat) ~ vec(kitten)
    - vodka – Russia + Mexico, teacher – school + hospital…

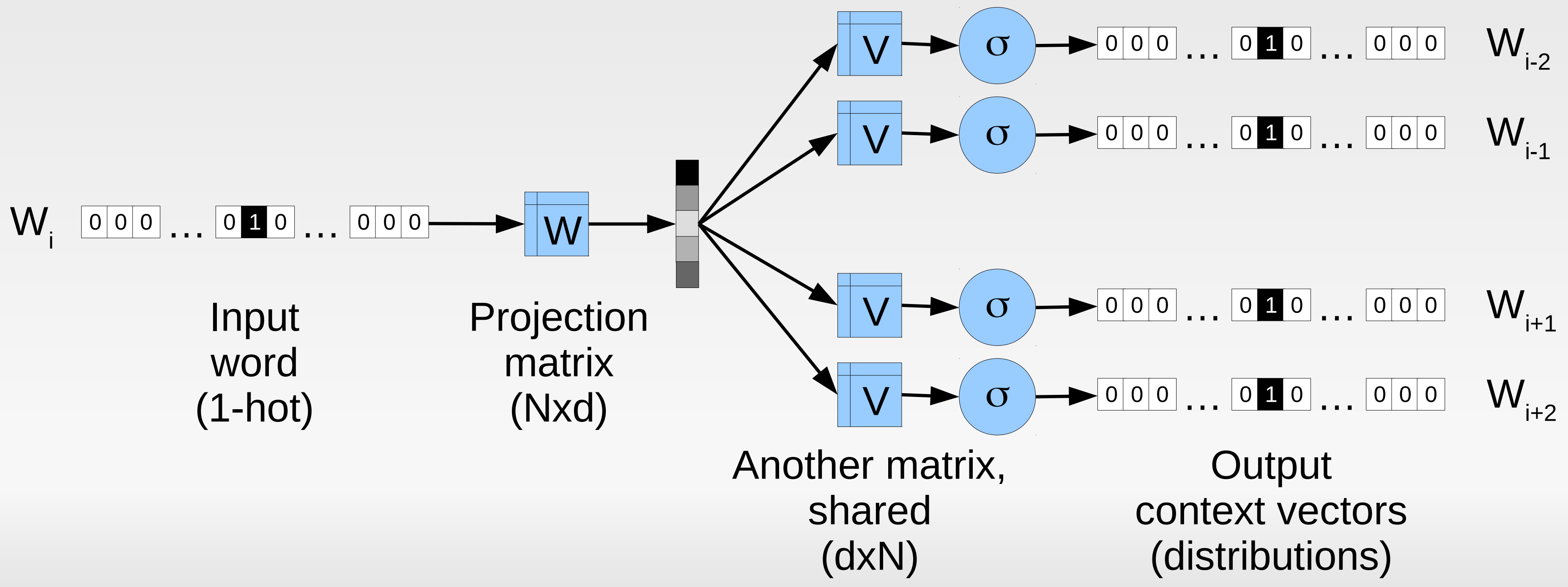
word2vec (Mikolov+, 2013)

- Predict word w_i from its context (CBOW)
  - E.g.: "*I had ______ for lunch*"
  - Sentence: … w_{i-2} w_{i-1} $\mathbf{w_i}$ w_{i+1} w_{i+2} …

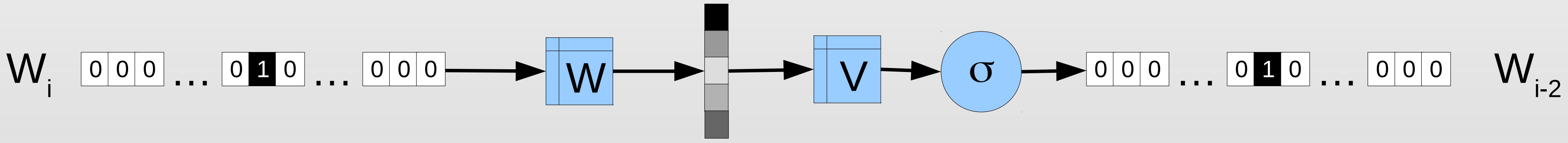


word2vec (Mikolov+, 2013)

- Predict context from a word w_i (SGNS)
  - E.g.: "_____ ______ *smelt* ______ ______"
  - Sentence: … $\mathbf{w_{i-2}}$ $\mathbf{w_{i-1}}$ w_i $\mathbf{w_{i+1}}$ $\mathbf{w_{i+2}}$ …



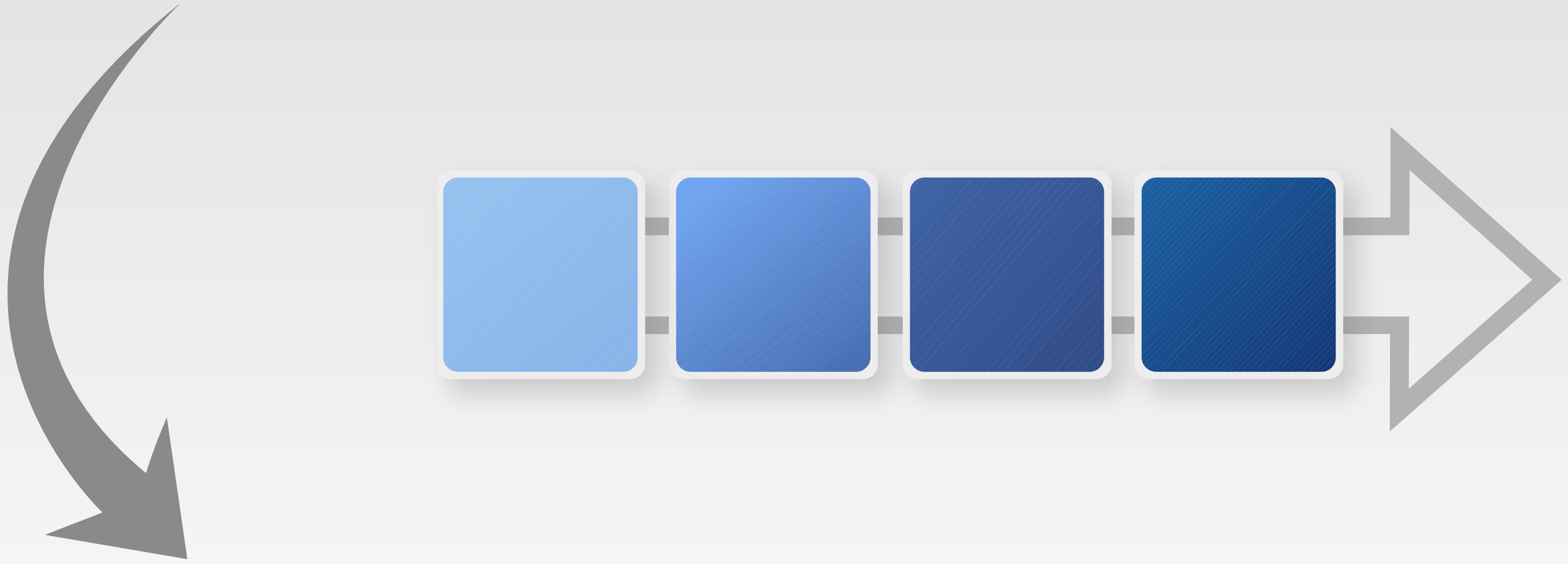
word2vec ~ implicit factorization



- Word embedding matrix $W \in \mathbf{R}^{Nxd}$
  - $\text{embedding}(\text{word}_i) = W[i] \in \mathbf{R}^d$
- Levy&Goldberg (2014)
  - word2vec SGNS implicitly factorizes M_{PMI}
  - $M_{PMI}[i, j] = \log [P(\text{word}_i \mid \text{context}_j) / P(\text{word}_i)]$
  - SGNS: $M_{PMI} = WV$
  - $M_{PMI} \in \mathbf{R}^{NxN} \rightarrow W \in \mathbf{R}^{Nxd}, V \in \mathbf{R}^{dxN}$

Problem 2: Sentences

Variable-length input sequences with long-distance relations between elements (sentences)



Fixed-sized neural units (attention mechanisms)

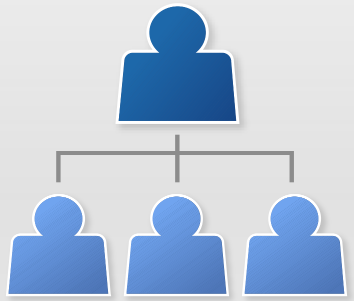
Processing sentences

- Convolutional neural netowrks
- Recurrent neural networks
- Attention mechanism
- Self-attentive networks

Convolutional neural networks

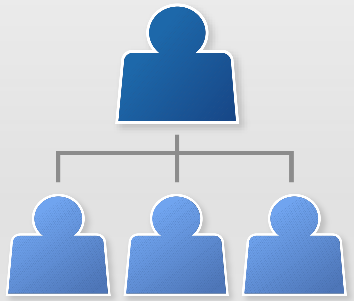
- Input: sequence of word embeddings
- Filters (size 3-5), norm, maxpooling

Convolutional neural networks



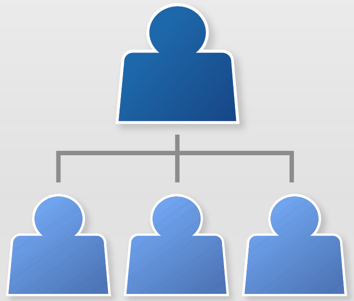
- Input: sequence of word embeddings
- Filters (size 3-5), norm, maxpooling
- Training deep CNNs hard → residual connections
  - Layer input averaged with output, skips non-linearity

Convolutional neural networks



- Input: sequence of word embeddings
- Filters (size 3-5), norm, maxpooling
- Training deep CNNs hard → residual connections
  - Layer input averaged with output, skips non-linearity
- Problem: capturing long-range dependencies
  - Receptive field of each filter is limited
  - *My <u>computer</u> works, but I have to buy a new <u>mouse</u>.*

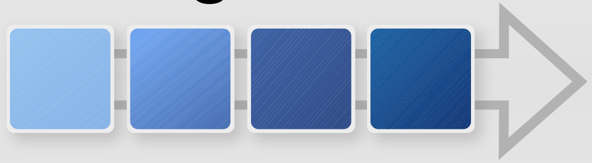
Convolutional neural networks



- Input: sequence of word embeddings
- Filters (size 3-5), norm, maxpooling
- Training deep CNNs hard → residual connections
  - Layer input averaged with output, skips non-linearity
- Problem: capturing long-range dependencies
  - Receptive field of each filter is limited
  - *My <u>computer</u> works, but I have to buy a new <u>mouse</u>.*
- Good for word *n*gram spotting
  - Sentiment analysis, named entity detection…

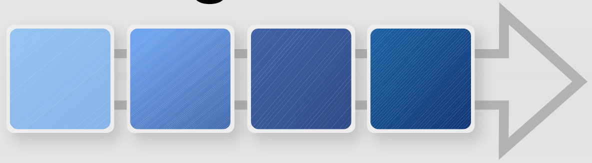
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN



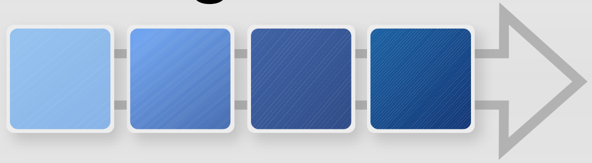
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN
- Problems



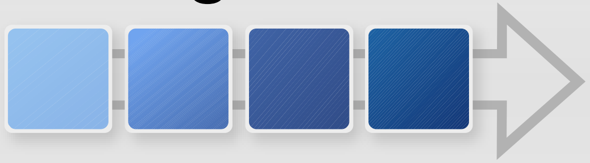
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN
- Problems
  - Vanishing gradient → memory cells (LSTM, GRU)



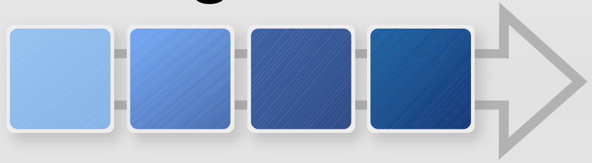
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN
- Problems
  - Vanishing gradient → memory cells (LSTM, GRU)
  - Long distance dependencies not perfectly captured



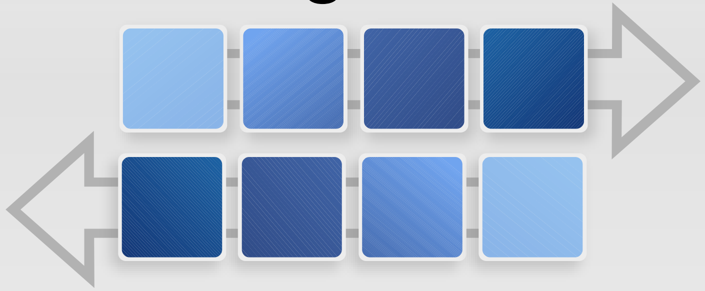
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN
- Problems
  - Vanishing gradient → memory cells (LSTM, GRU)
  - Long distance dependencies not perfectly captured
  - Final state is biased (“forgetting”)
    - …sentence end better captured than sentence start



Recurrent neural networks

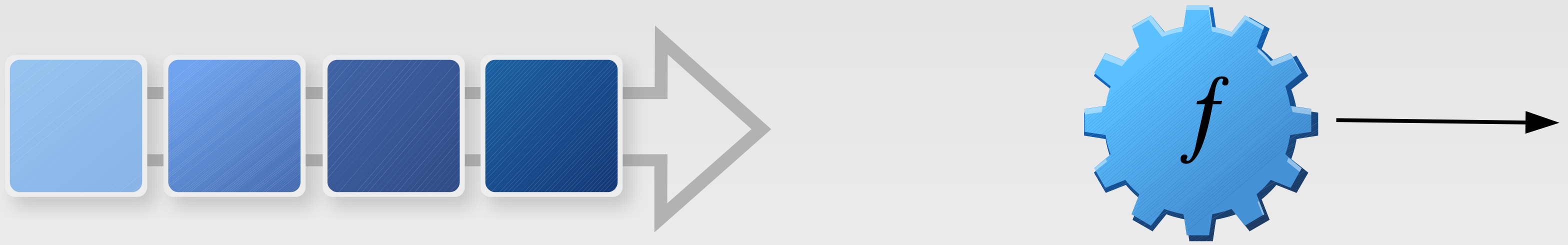
- Input: sequence of word embeddings
- Output: final state of RNN
- Problems
  - Vanishing gradient → memory cells (LSTM, GRU)
  - Long distance dependencies not perfectly captured
  - Final state is biased ("forgetting")
    - …sentence end better captured than sentence start
    - Bidirectional RNN, output = concat of both final states
      - Still may not well capture the middle parts…



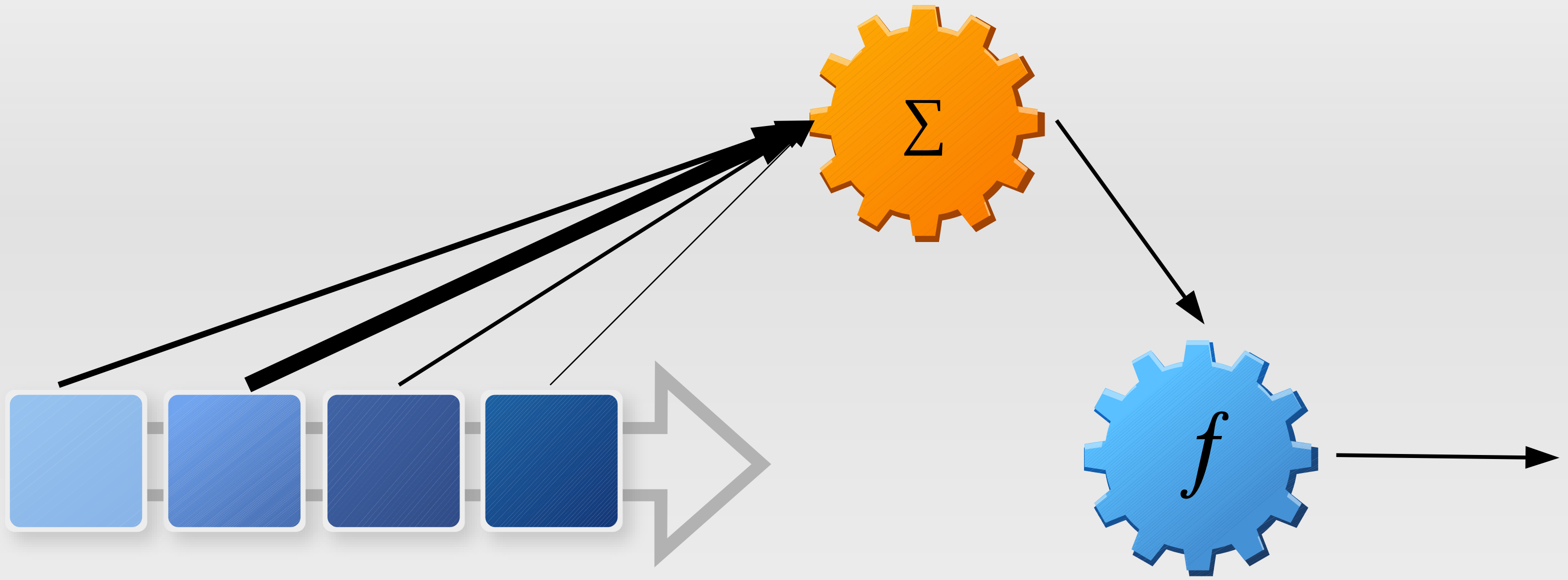
Recurrent neural networks

- Input: sequence of word embeddings
- Output: final state of RNN
- Problems
  - Vanishing gradient → memory cells (LSTM, GRU)
  - Long distance dependencies not perfectly captured
  - Final state is biased (“forgetting”)
    - …sentence end better captured than sentence start
    - Bidirectional RNN, output = concat of both final states
      - Still may not well capture the middle parts…
    - Using all hidden states as output, not just the final one
      - We loose the fixed-sized representation

Attention (on top of a RNN)

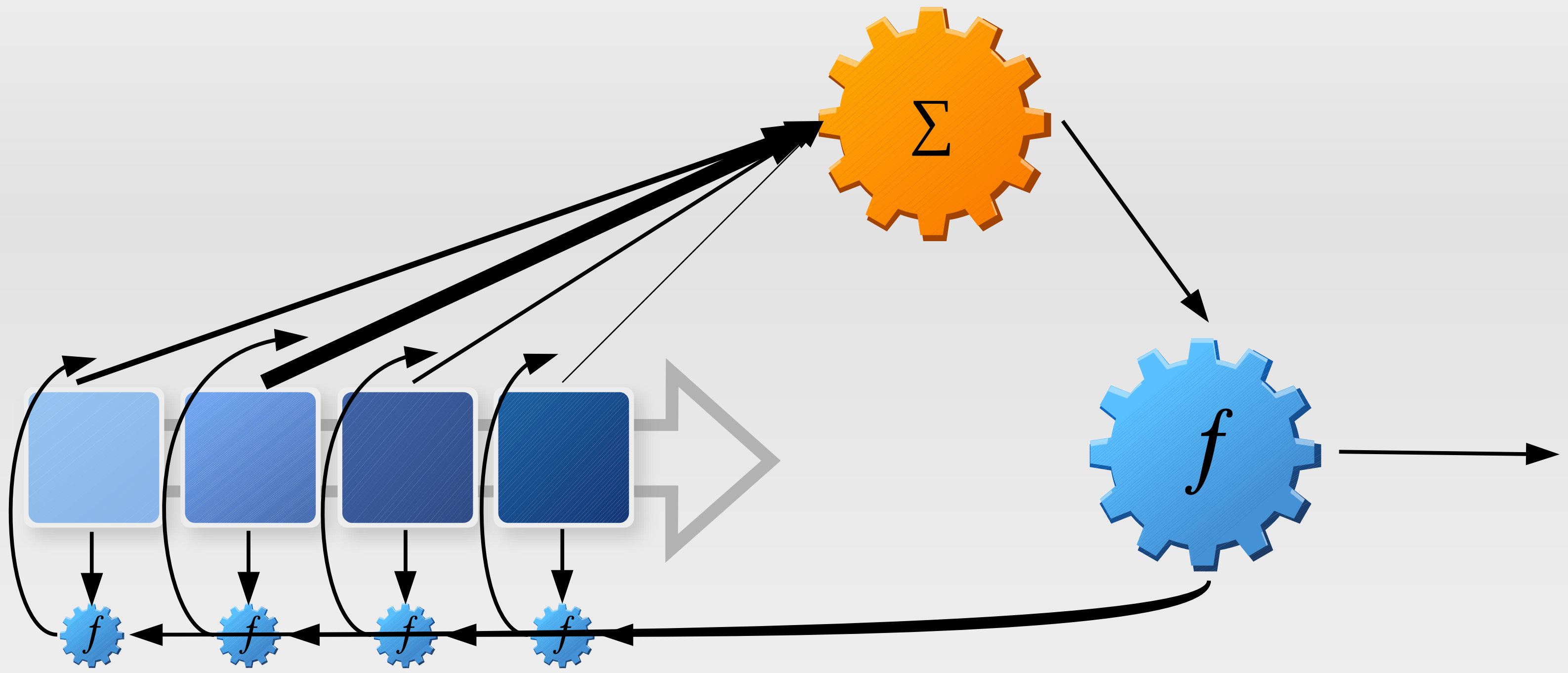


Attention (on top of a RNN)



- Classifier/decoder gets a fixed-size context vector
  - Weighted average of encoder hidden states

Attention (on top of a RNN)



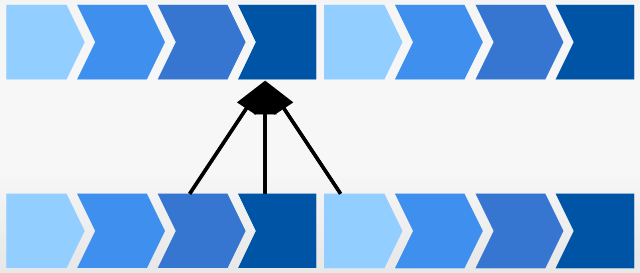
- Classifier/decoder gets a fixed-size context vector
  - Weighted average of encoder hidden states
  - Attention weights computed by a feed-forward subnet
    - weight$_i$ ~ NN(state$_i$, state$_{decoder}$)

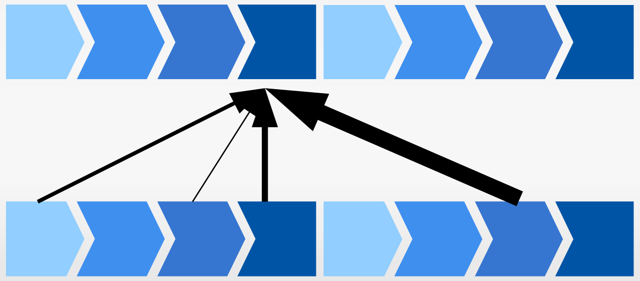
Advanced attention

- Multi-head attention
  - Multiple attention heads (~8), each has its own distro
  - Resulting context vectors concatenated

Advanced attention

- Multi-head attention
  - Multiple attention heads (~8), each has its own distro
  - Resulting context vectors concatenated
- Self-attentitive encoder (SAN, Transformer)
  - CNN/attention hybrid
  - CNN: cell gets small local context via filters
  - SAN: cell gets global context via attention heads





Conclusion

Conclusion

- Words → word embeddings
  - Too many, too sparse
  - Word meaning ~ context in which it appears
  - Cooccurrence matrix, implicit/explicit factorization

Conclusion

- Words → word embeddings
  - Too many, too sparse
  - Word meaning ~ context in which it appears
  - Cooccurrence matrix, implicit/explicit factorization
- Sentences → attention
  - Variable length, complex internal structure
  - biRNN (LSTM, GRU), CNN+residuals
  - Attention: weighted sum of encoder hidden states
  - Self-attention: à la CNN, filters → attention

References

- **Word embeddings:**
  - **Distributional hyp.:** Harris: *Distributional structure.* Word, 1954
  - **First:** Bengio+: *A neural probabilistic language model.* JMLR, 2003
  - **Efficient implicit (word2vec):** Mikolov+: *Linguistic Regularities in Continuous Space Word Representations.* NAACL, 2013
  - **Explicit (TSVD):** Levy&Goldberg: *Neural Word Embedding as Implicit Matrix Factorization.* NIPS, 2014
- **Recurrent neural networks and attention:**
  - **LSTM:** Hochreiter+: *Long short-term memory.* NeCo, 1997
  - **Attention:** Bahdanau+: *Neural Machine Translation by Jointly Learning to Align and Translate.* CoRR, 2014
  - **Tranformer SAN:** Vaswani+: *Attention is all you need.* NIPS, 2017

Thank you for your attention

Rudolf Rosa
rosa@ufal.mff.cuni.cz

Deep Neural Networks
in Natural Language Processing

Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

http://ufal.mff.cuni.cz/rudolf-rosa/