Neural networks

Natural language processing - motivation

NATURAL LANGUAGE PROCESSING

Topics: natural language processing (NLP)

- Natural language processing is concerned with tasks involving language data
  - we will focus on text data NLP
  - speech processing is also NLP, though it has its own dedicated research community

- Much like for computer vision, we can design neural networks specifically adapted to the processing of text data
  - main issue: text data is inherently high dimensional

Neural networks

Natural language processing - preprocessing

NATURAL LANGUAGE PROCESSING

Topics: tokenization

- Typical preprocessing steps of text data
  - tokenize text (from a long string to a list of token strings)

“He’s spending 7 days in San Francisco.”

“He”
“’s”
“spending”
“7”
“days”
“in”
“San Francisco”
“.”

  - for many datasets, this has already been done for you
  - splitting into tokens based on spaces and separating punctuation is good enough in English or French

NATURAL LANGUAGE PROCESSING

Topics: lemmatization

- Typical preprocessing steps of text data
  - lemmatize tokens (put into standard form)

“He”	“he”
“’s”	“be”
“spending”	“spend”
“7”	”NUMBER”
“days”	“day”
“in”	“in”
“San Francisco”	“San Francisco”
“.”	“.”

  - the specific lemmatization will depend on the problem we want to solve
    - we can remove variations of words that are not relevant to the task at hand

NATURAL LANGUAGE PROCESSING

Topics: vocabulary

- Typical preprocessing steps of text data
  - form vocabulary of words that maps lemmatized words to a unique ID (position of word in vocabulary)
  - different criteria can be used to select which words are part of the vocabulary
    - pick most frequent words
    - ignore uninformative words from a user-defined short list (ex.: " `the` ", " `a` ", etc.)
  - all words not in the vocabulary will be mapped to a special "out-of-vocabulary" ID

- Typical vocabulary sizes will vary between 10 000 and 250 000

NATURAL LANGUAGE PROCESSING

Topics: vocabulary

- Example:

Vocabulary

Word	w
"`the`"	1
"`and`"	2
"`dog`"	3
"`.`"	4
"`OOV`"	5

"`the`"	1
"`cat`"	5
"`and`"	2
"`the`"	1
"`dog`"	3
"`play`"	5
"`.`"	4

- We will note word IDs with the symbol w
  - can think of w as a categorical feature for the original word
  - we will sometimes refer to w as a word, for simplicity

Neural networks

Natural language processing - one-hot encoding

NATURAL LANGUAGE PROCESSING

Topics: one-hot encoding

- From its word ID, we get a basic representation of a word through the one-hot encoding of the ID
  - the one-hot vector of an ID is a vector filled with 0s, except for a 1 at the position associated with the ID
    - ex.: for vocabulary size D=10, the one-hot vector of word ID w=4 is

$$\mathrm{e}(w) = [\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ ]$$

  - a one-hot encoding makes no assumption about word similarity
    - $||\mathrm{e}(w) - \mathrm{e}(w')||^2 = 0$ if $w = w'$
    - $||\mathrm{e}(w) - \mathrm{e}(w')||^2 = 2$ if $w \neq w'$
    - all words are equally different from each other
  - this is a natural representation to start with, though a poor one

NATURAL LANGUAGE PROCESSING

Topics: one-hot encoding

- The major problem with the one-hot representation is that it is very high-dimensional
  - the dimensionality of $\mathrm{e}(w)$ is the size of the vocabulary
  - a typical vocabulary size is $\approx 100\ 000$
  - a window of 10 words would correspond to an input vector of at least $1\ 000\ 000$ units!
- This has 2 consequences:
  - vulnerability to overfitting
    - millions of inputs means millions of parameters to train in a regular neural network
  - computationally expensive
    - not all computations can be sparsified (ex.: reconstruction in autoencoder)

Neural networks

Natural language processing - word representations

NATURAL LANGUAGE PROCESSING

Topics: one-hot encoding

- The major problem with the one-hot representation is that it is very high-dimensional
  - the dimensionality of $\mathrm{e}(w)$ is the size of the vocabulary
  - a typical vocabulary size is $\approx 100\ 000$
  - a window of 10 words would correspond to an input vector of at least $1\ 000\ 000$ units!
- This has 2 consequences:
  - vulnerability to overfitting
    - millions of inputs means millions of parameters to train in a regular neural network
  - computationally expensive
    - not all computations can be sparsified (ex.: reconstruction in autoencoder)

WORD REPRESENTATIONS

Topics: continuous word representation

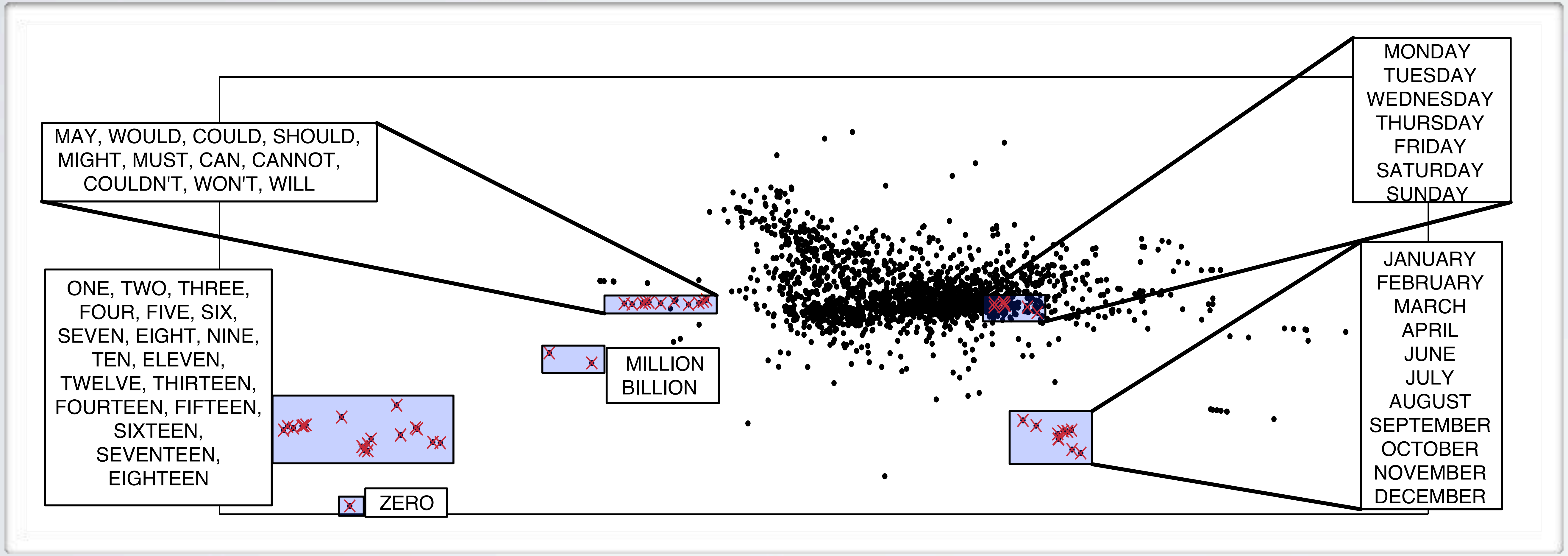
- Idea: learn a continuous representation of words
  - each word w is associated with a real-valued vector $C(w)$

Word	w	$C(w)$
" the "	1	[0.6762, -0.9607, 0.3626, -0.2410, 0.6636]
" a "	2	[0.6859, -0.9266, 0.3777, -0.2140, 0.6711]
" have "	3	[0.1656, -0.1530, 0.0310, -0.3321, -0.1342]
" be "	4	[0.1760, -0.1340, 0.0702, -0.2981, -0.1111]
" cat "	5	[0.5896, 0.9137, 0.0452, 0.7603, -0.6541]
" dog "	6	[0.5965, 0.9143, 0.0899, 0.7702, -0.6392]
" car "	7	[-0.0069, 0.7995, 0.6433, 0.2898, 0.6359]
...	...	...

WORD REPRESENTATIONS

Topics: continuous word representation

- Idea: learn a continuous representation of words
  - we would like the distance $||C(w)\text{-}C(w')||$ to reflect meaningful similarities between words



(from Blitzer et al. 2004)

WORD REPRESENTATIONS

Topics: continuous word representation

- Idea: learn a continuous representation of words
  - we could then use these representations as input to a neural network
  - to represent a window of 10 words $[w_1, \ldots, w_{10}]$, we concatenate the representations of each word

$$\mathbf{x} = [C(w_1)^\top, \ldots, C(w_{10})^\top]^\top$$

- We learn these representations by gradient descent
  - we don't only update the neural network parameters
  - we also update each representation $C(w)$ in the input $\mathbf{x}$ with a gradient step

$$C(w) \Longleftarrow C(w) - \alpha \nabla_{C(w)} l$$

where l is the loss function optimized by the neural network

WORD REPRESENTATIONS

Topics: word representations as a lookup table

- Let $\mathbf{C}$ be a matrix whose rows are the representations $C(w)$
  - obtaining $C(w)$ corresponds to the multiplication $\mathrm{e}(w)^\top \mathbf{C}$
  - view differently, we are projecting $\mathrm{e}(w)$ onto the columns of C
    - this is a reduction of the dimensionality of the one-hot representations $\mathrm{e}(w)$
  - this is a continuous transformation, through which we can propagate gradients

- In practice, we implement $C(w)$ with a lookup table, not with a multiplication
  - $C(w)$ returns an array pointing to the w^{th} row of $\mathbf{C}$

Neural networks

Natural language processing - language modeling

LANGUAGE MODELING

Topics: language modeling

- A language model is a probabilistic model that assigns probabilities to any sequence of words

$$p(w_1,\ \ldots\ ,w_T)$$

  - language modeling is the task of learning a language model that assigns high probabilities to well formed sentences
  - plays a crucial role in speech recognition and machine translation systems

“`a person smart`”

“`une personne intelligente`” ?

“`a smart person`”

LANGUAGE MODELING

Topics: language modeling

- An assumption frequently made is the n^{th} order Markov assumption

$$p(w_1, \ldots ,w_T) = \prod_{t=1}^{T} p(w_t \mid w_{t-(n-1)} , \ldots ,w_{t-1})$$

  - the t^{th} word was generated based only on the $n{-}1$ previous words
  - we will refer to $w_{t-(n-1)} , \ldots ,w_{t-1}$ as the context

LANGUAGE MODELING

Topics: n-gram model

- An n-gram is a sequence of n words
  - unigrams (n=1): "**is**", "**a**", "**sequence**", etc.
  - bigrams (n=2): ["**is**", "**a**"], ["**a**", "**sequence**"], etc.
  - trigrams (n=3): ["**is**", "**a**", "**sequence**"], ["**a**", "**sequence**", "**of**"], etc.
- n-gram models estimate the conditional from n-grams counts

$$p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1}) = \frac{\text{count}(w_{t-(n-1)}, \ldots, w_{t-1}, w_t)}{\text{count}(w_{t-(n-1)}, \ldots, w_{t-1}, \cdot)}$$

  - the counts are obtained from a training corpus (a data set of word text)

LANGUAGE MODELING

Topics: n-gram model

- Issue: data sparsity
  - we want n to be large, for the model to be realistic
  - however, for large values of n, it is likely that a given n-gram will not have been observed in the training corpora
  - smoothing the counts can help
    - combine $\mathrm{count}(w_1\ ,\ w_2\ ,\ w_3\ ,\ w_4)$, $\mathrm{count}(w_2\ ,\ w_3\ ,\ w_4)$, $\mathrm{count}(w_3\ ,\ w_4)$, and $\mathrm{count}(w_4)$ to estimate $p(w_4\,|w_1,\ w_2,\ w_3)$
  - this only partly solves the problem

Neural networks

Natural language processing - neural network language model

LANGUAGE MODELING

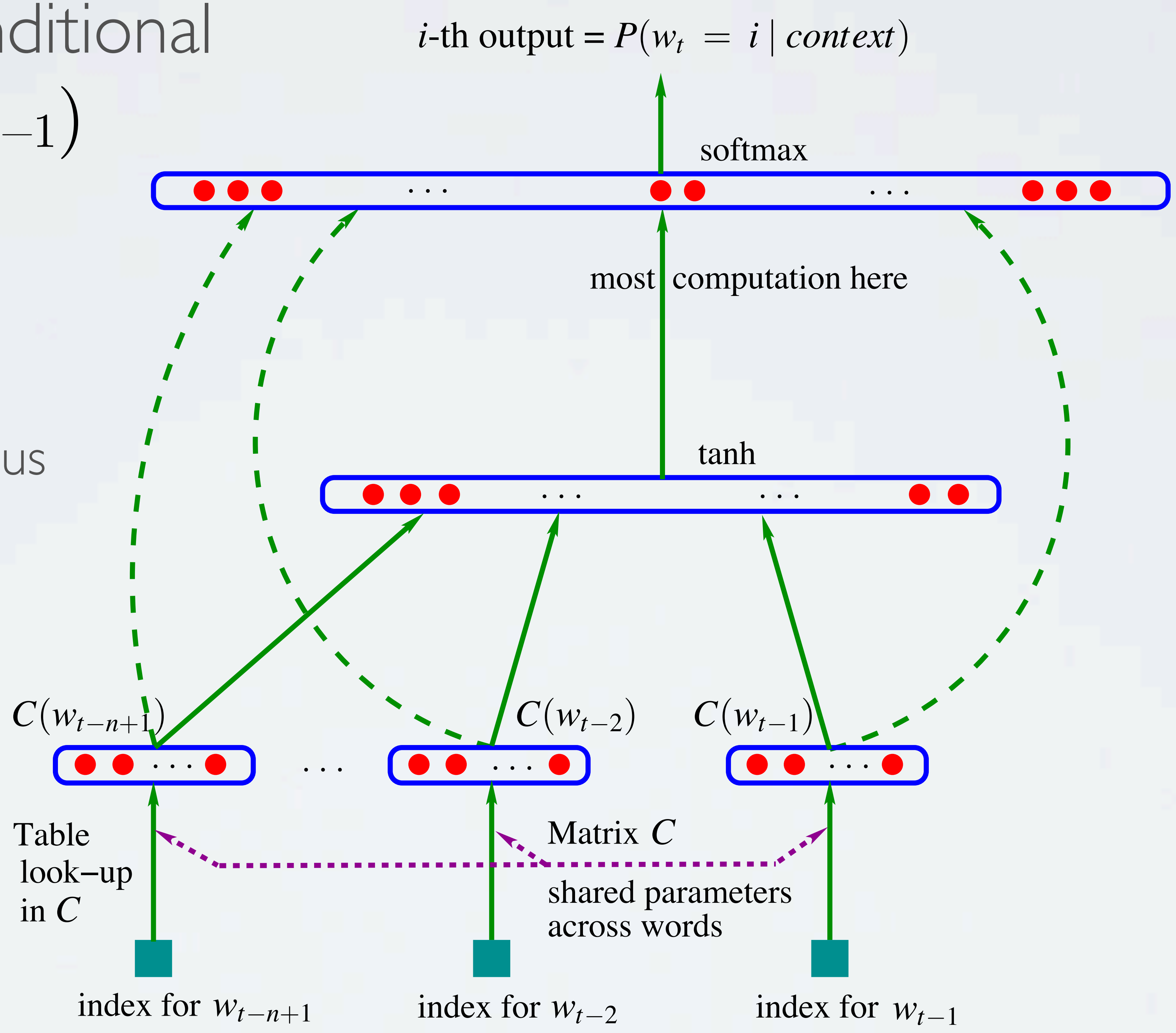
Topics: n-gram model

- Issue: data sparsity
  - we want n to be large, for the model to be realistic
  - however, for large values of n, it is likely that a given n-gram will not have been observed in the training corpora
  - smoothing the counts can help
    - combine $\mathrm{count}(w_1, w_2, w_3, w_4)$, $\mathrm{count}(w_2, w_3, w_4)$, $\mathrm{count}(w_3, w_4)$, and $\mathrm{count}(w_4)$ to estimate $p(w_4 \mid w_1, w_2, w_3)$
  - this only partly solves the problem

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus

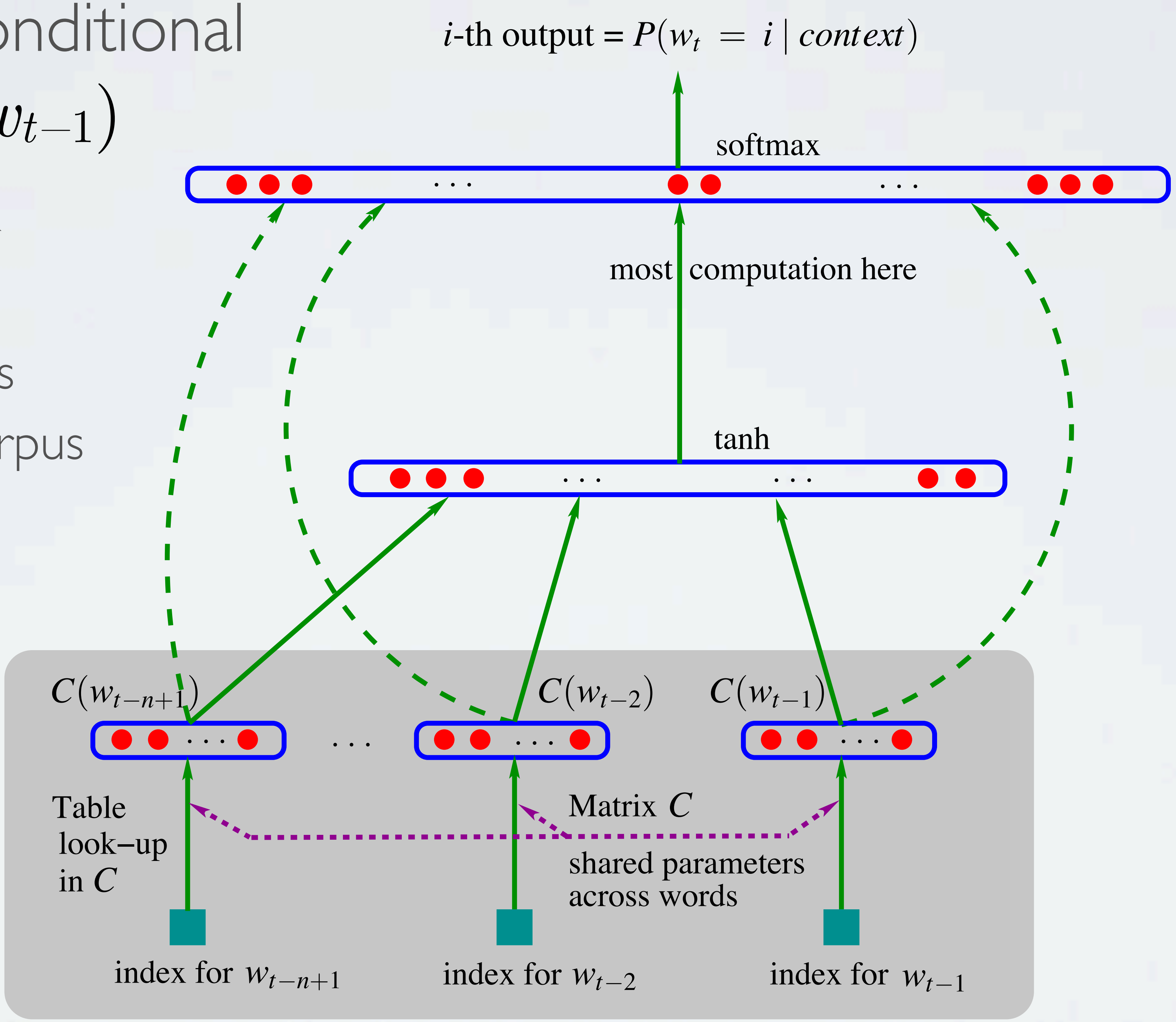


Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus

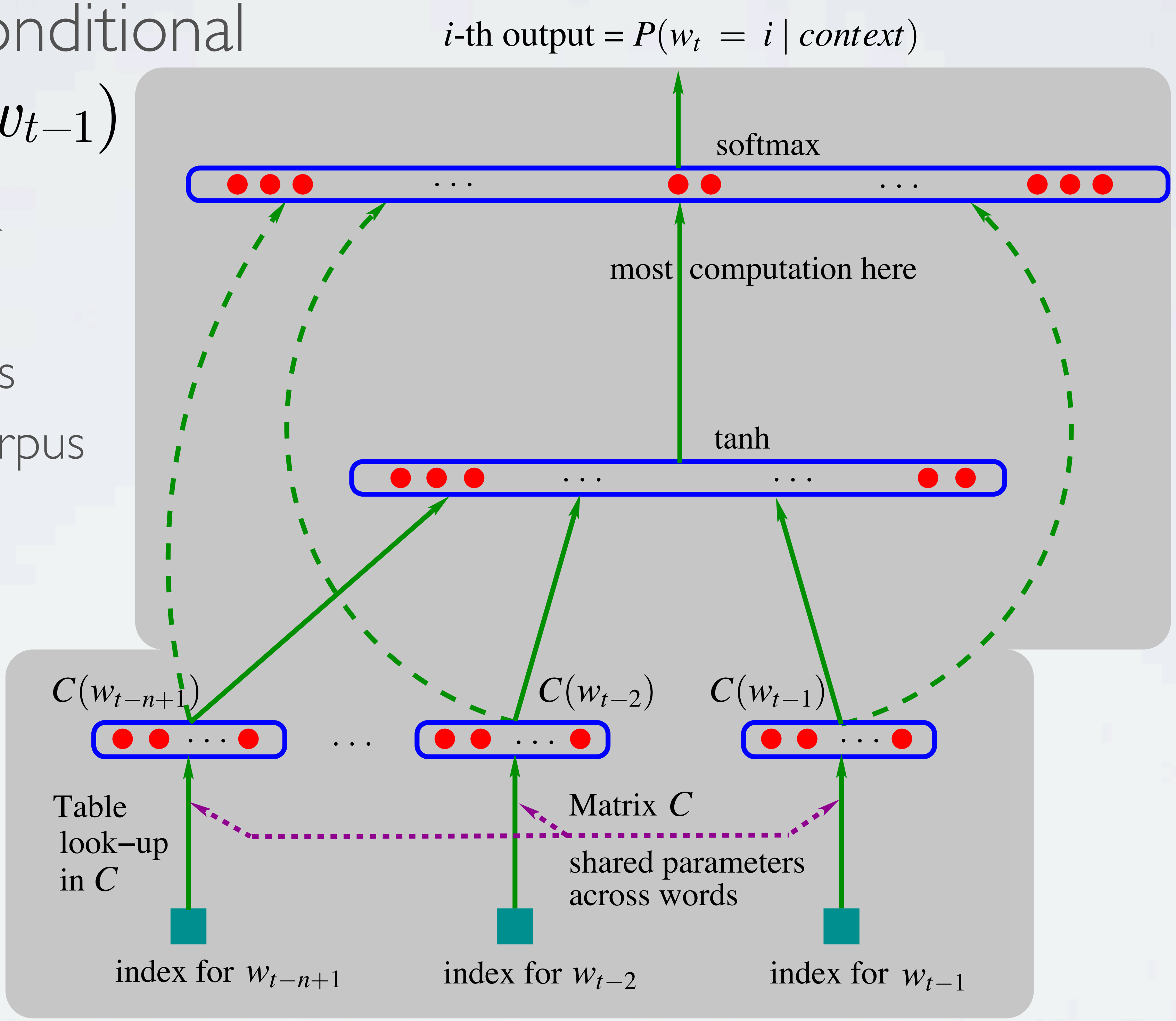


Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus

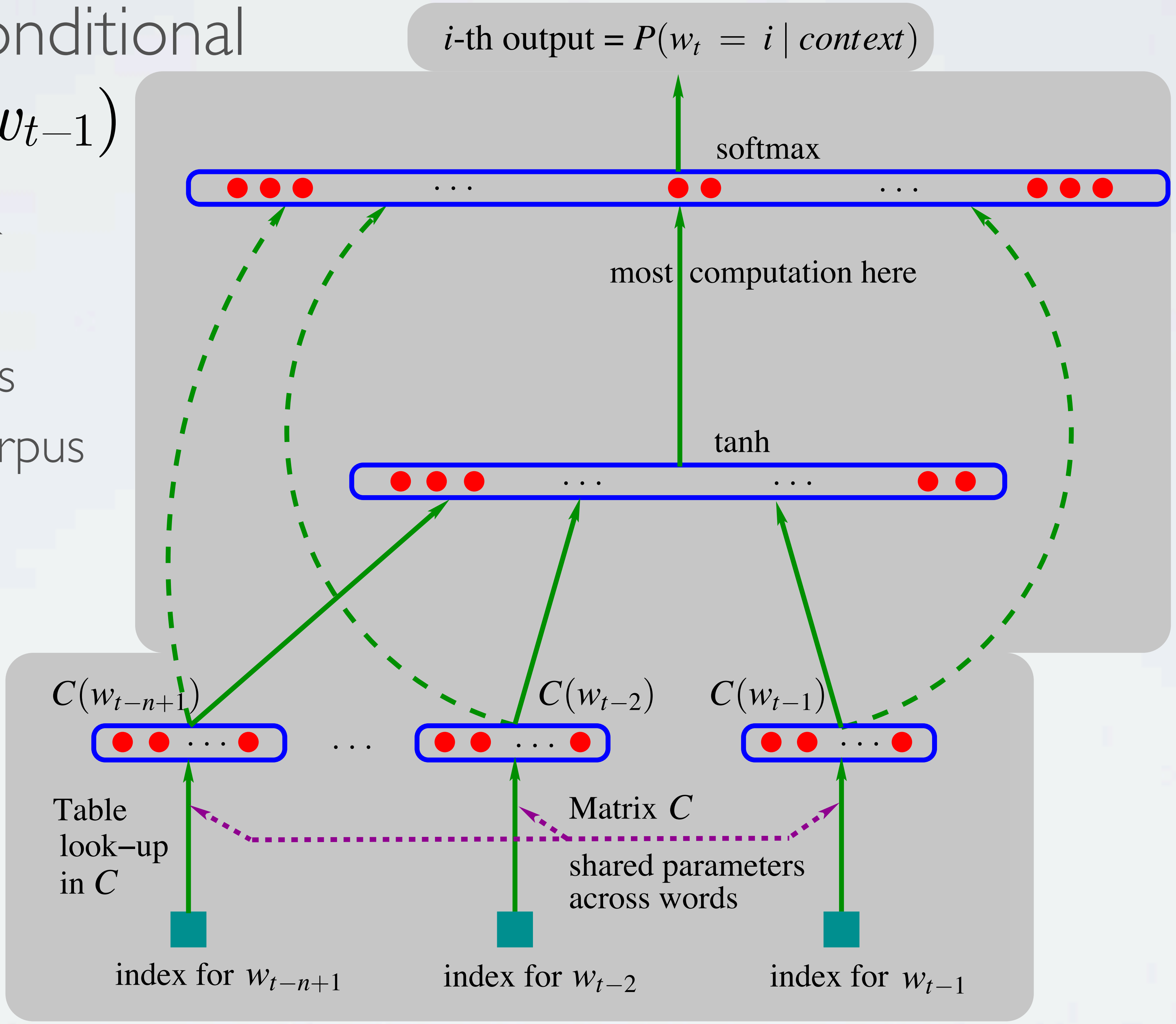


Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus



Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Can potentially generalize to contexts not seen in training set
  - example: $p(\text{“}\,\texttt{eating}\,\text{”} \mid \text{“}\,\texttt{the}\,\text{”}, \text{“}\,\texttt{cat}\,\text{”}, \text{“}\,\texttt{is}\,\text{”})$
    - imagine 4-gram [“ **the** ”,“ **cat** ”,“ **is** ”, “ **eating** ”] is not in training corpus, but [“ **the** ”,“ **dog** ”,“ **is** ”, “ **eating** ”] is
    - if the word representations of “ **cat** ” and “ **dog** ” are similar, then the neural network will be able to generalize to the case of “ **cat** ”
    - neural network could learn similar word representations for those words based on other 4-grams:

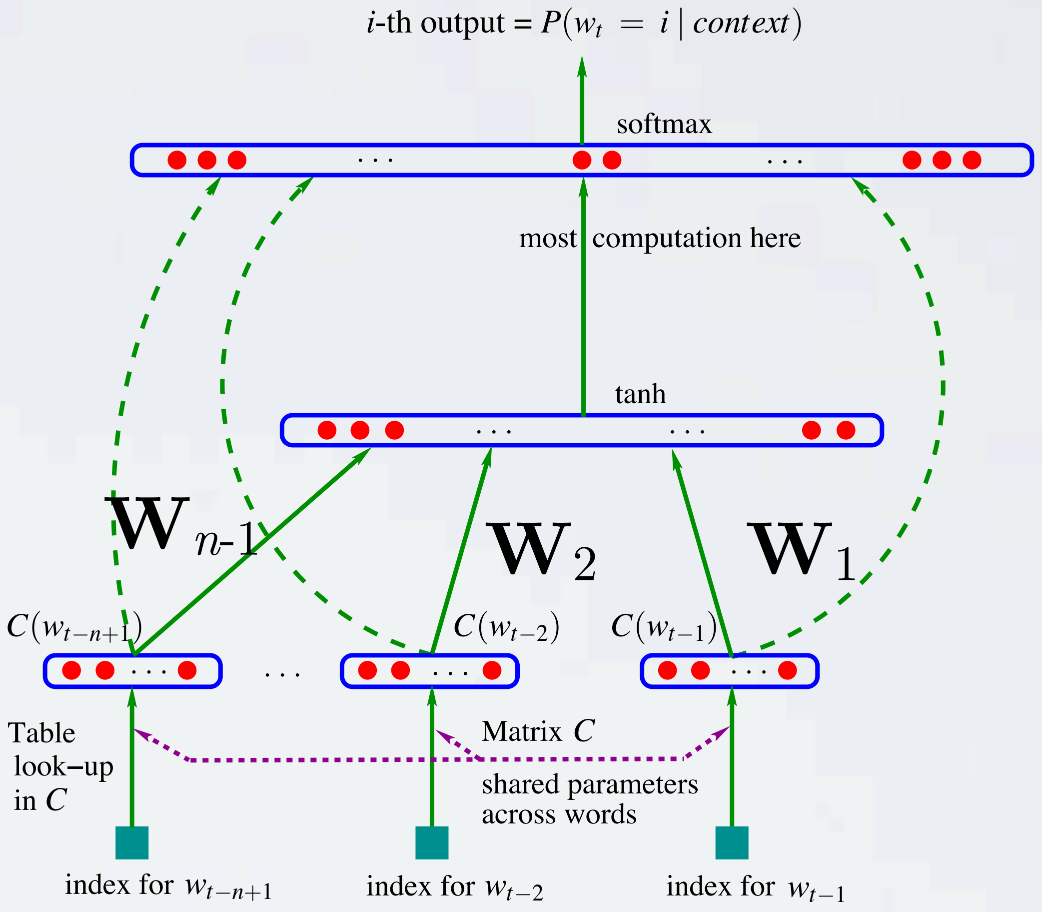
      [“ the ”,“ cat ”,“ was ”, “ sleeping ”]
      [“ the ”,“ dog ”,“ was ”, “ sleeping ”]

NEURAL NETWORK LANGUAGE MODEL

Topics: word representation gradients

- We know how to propagate gradients in such a network
  - we know how to compute the gradient for the linear activation of the hidden layer $\nabla_{\mathbf{a}(\mathbf{x})} l$
  - let's note the submatrix connecting w_{t-i} and the hidden layer as $\mathbf{W}_i$
- The gradient wrt $C(w)$ for any w is

$$\nabla_{C(w)} l = \sum_{i=1}^{n-1} 1_{(w_{t-i}=w)} \, \mathbf{W}_i^\top \, \nabla_{\mathbf{a}(\mathbf{x})} l$$



Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: word representation gradients

- Example: $[\text{“}\,\mathtt{the}\,\text{”}, \text{“}\,\mathtt{dog}\,\text{”}, \text{“}\,\mathtt{and}\,\text{”}, \text{“}\,\mathtt{the}\,\text{”}, \text{“}\,\mathtt{cat}\,\text{”}]$

$$\begin{matrix} w_3 & w_4 & w_5 & w_6 & w_7 \\ \| & \| & \| & \| & \\ 21 & 3 & 14 & 21 & \end{matrix}$$

  - the loss is $l = -\log p(\text{“}\,\mathtt{cat}\,\text{”} \,|\, \text{“}\,\mathtt{the}\,\text{”}, \text{“}\,\mathtt{dog}\,\text{”}, \text{“}\,\mathtt{and}\,\text{”}, \text{“}\,\mathtt{the}\,\text{”})$
  - $\nabla_{C(3)} l = \mathbf{W}_3^\top \nabla_{\mathbf{a}(\mathbf{x})} l$
  - $\nabla_{C(14)} l = \mathbf{W}_2^\top \nabla_{\mathbf{a}(\mathbf{x})} l$
  - $\nabla_{C(21)} l = \mathbf{W}_1^\top \nabla_{\mathbf{a}(\mathbf{x})} l + \mathbf{W}_4^\top \nabla_{\mathbf{a}(\mathbf{x})} l$
  - $\nabla_{C(w)} l = 0$ for all other words w

- Only need to update the representations $C(3)$, $C(14)$ and $C(21)$,

NEURAL NETWORK LANGUAGE MODEL

Topics: performance evaluation

- In language modeling, a common evaluation metric is the perplexity
  - it is simply the exponential of the average negative log-likelihood
- Evaluation on Brown corpus
  - n-gram model (Kneser-Ney smoothing): **321**
  - neural network language model: **276**
  - neural network + n-gram: **252**

Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: performance evaluation

- A more interesting (and less straightforward) way of evaluating a language model is within a particular application
  - does a language model improve the performance of a machine translation or speech recognition system

- Later work has shown improvements in both cases
  - Connectionist language modeling for large vocabulary continuous speech recognition
    Schwenk and Gauvain, 2002
  - Continuous-Space Language Models for Statistical Machine Translation
    Schwenk, 2010

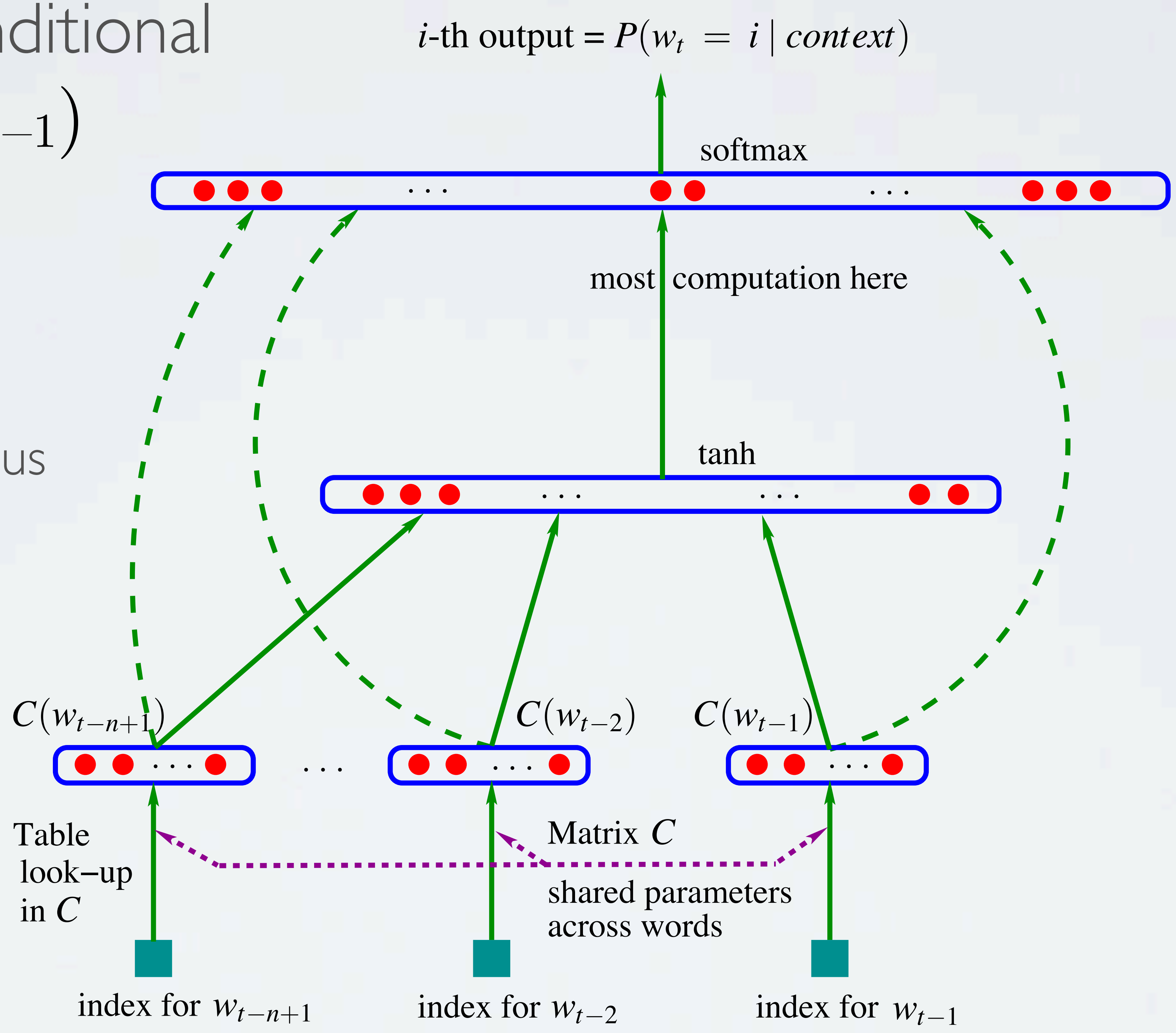
Neural networks

Natural language processing - hierarchical output layer

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus

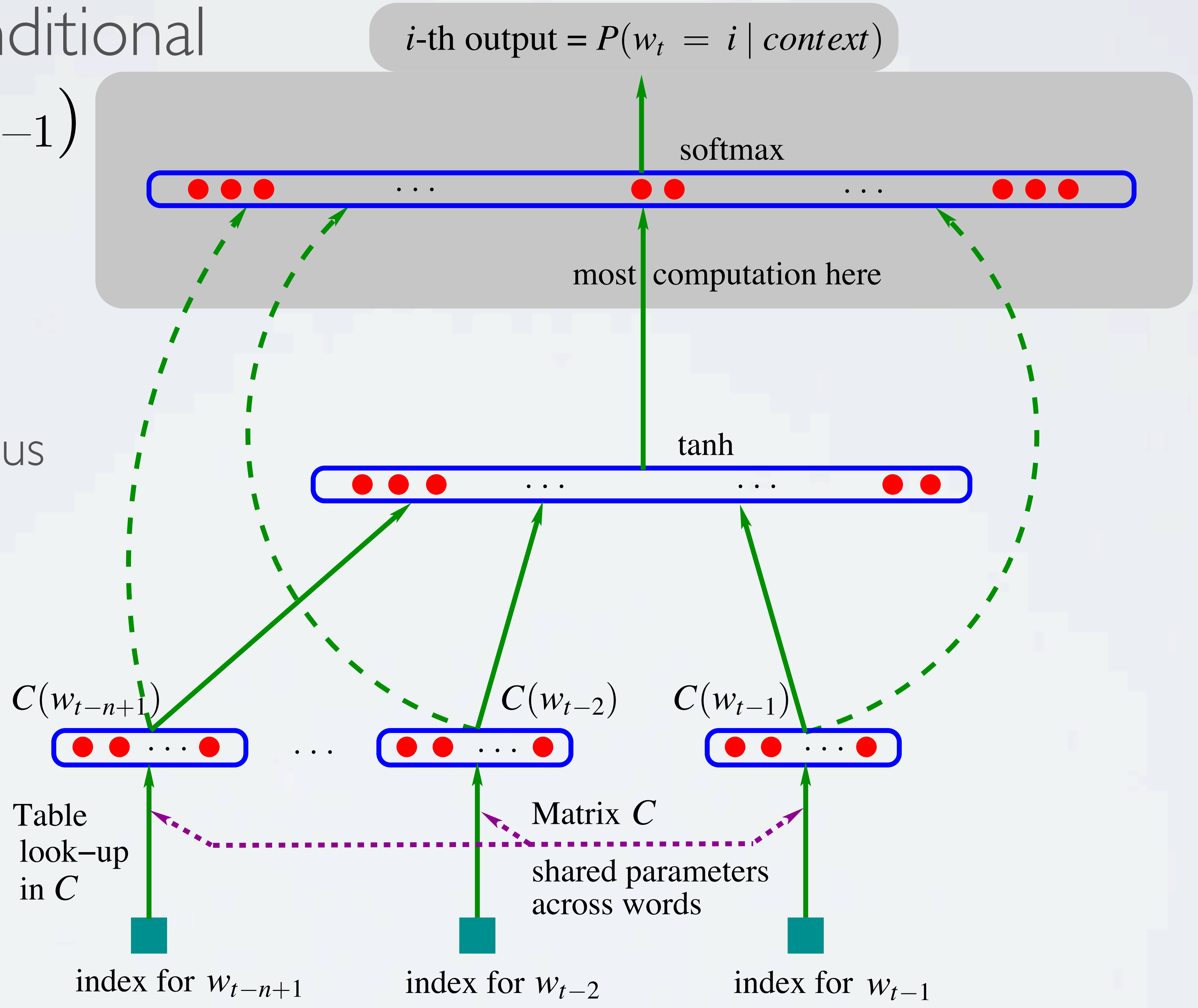


Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: neural network language model

- Solution: model the conditional
  $p(w_t \mid w_{t-(n-1)}, \ldots, w_{t-1})$
  with a neural network
  - learn word representations to allow transfer to n-grams not observed in training corpus



Bengio, Ducharme, Vincent and Jauvin, 2003

NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

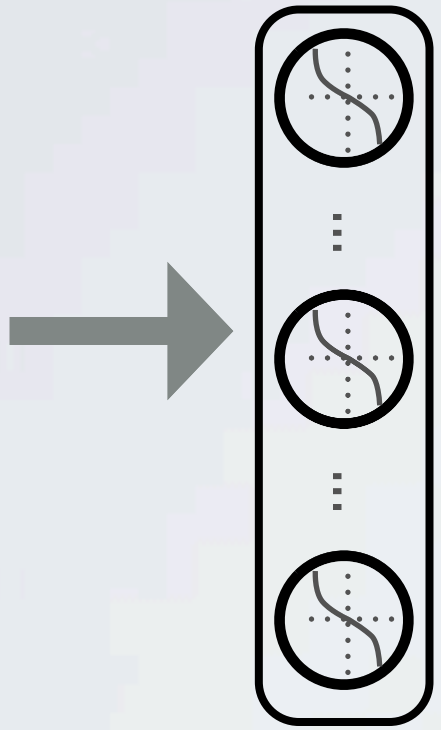
- Example: [" **the** "," **dog** "," **and** ", " **the** ", " **cat** "]

" **the** "

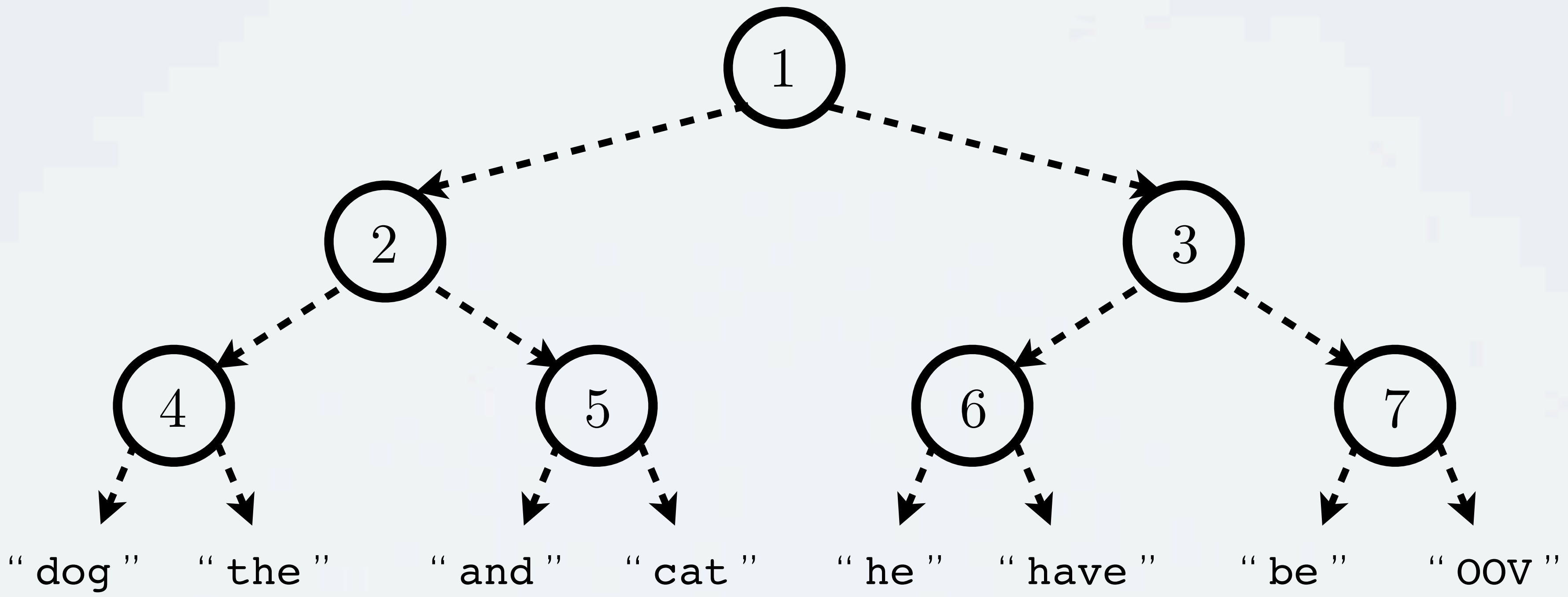
" **dog** "

" **and** "

" **the** "



$p($" **cat** "$\mid \text{context}) =$

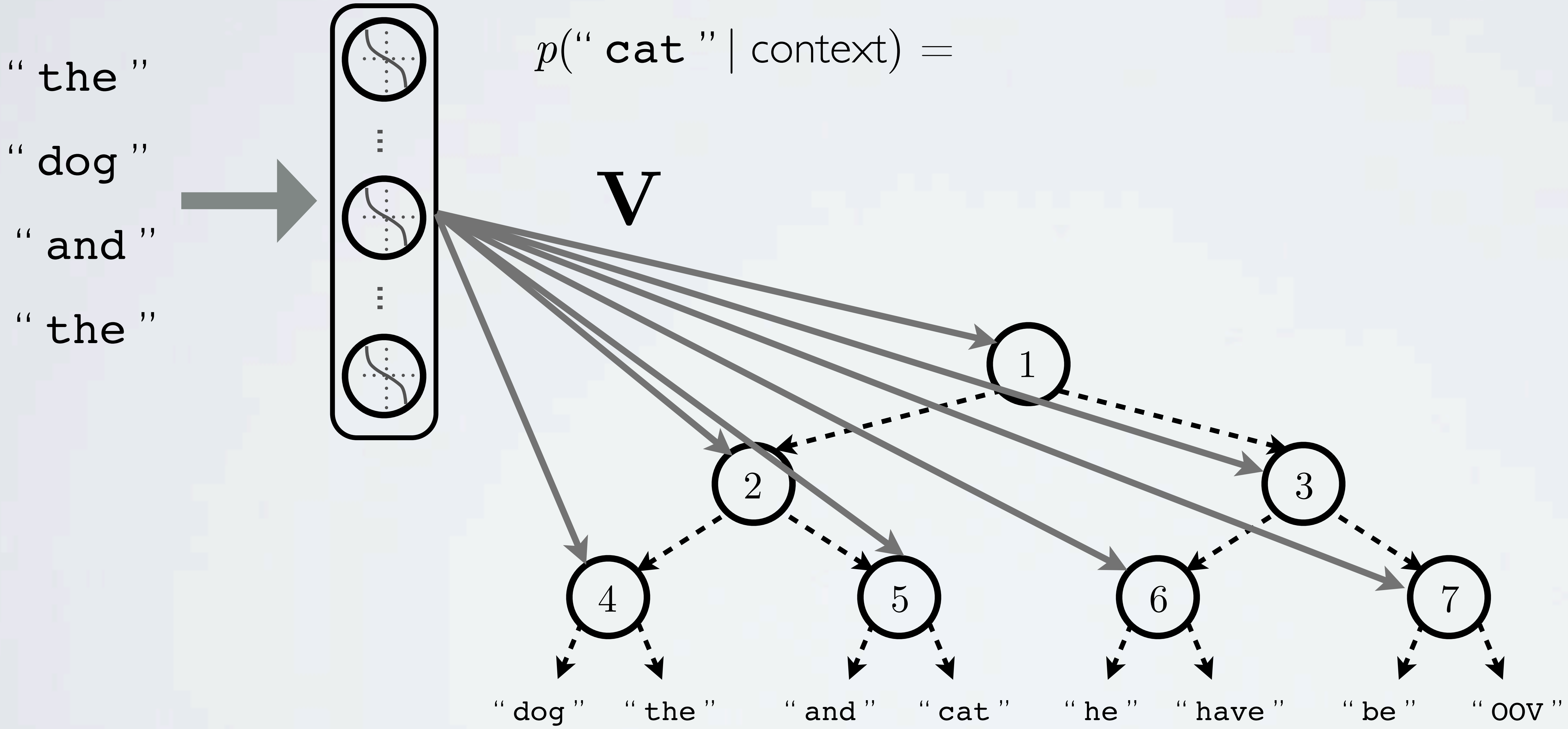


NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

- Example: [" **the** "," **dog** "," **and** ", " **the** ", " **cat** "]

$p($" **cat** "$\mid$ context$) =$

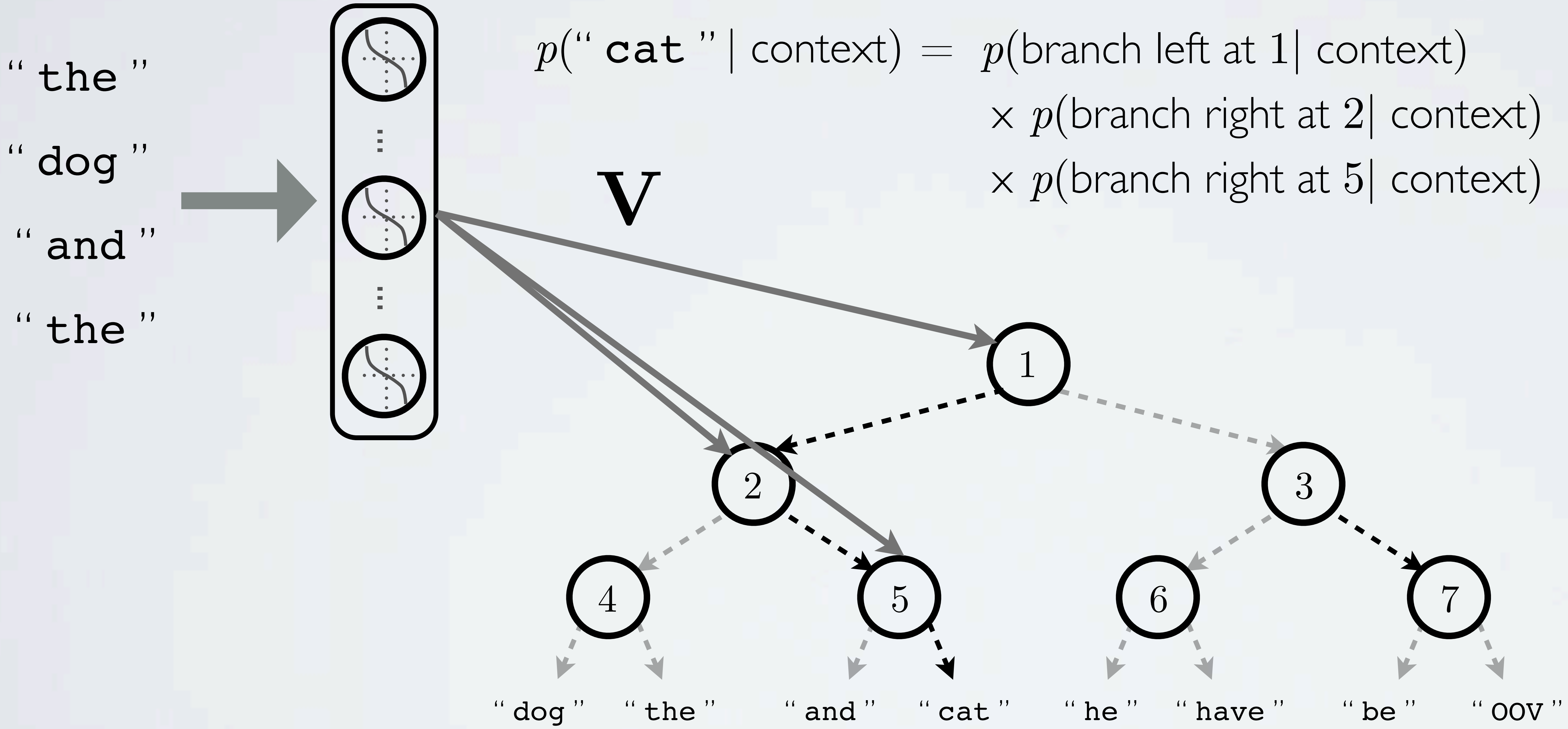


NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

- Example: [" **the** "," **dog** "," **and** ", " **the** ", " **cat** "]

$$p(\text{“}\,\mathtt{cat}\,\text{”}\,|\,\text{context}) = p(\text{branch left at } 1|\text{ context}) \times p(\text{branch right at } 2|\text{ context}) \times p(\text{branch right at } 5|\text{ context})$$

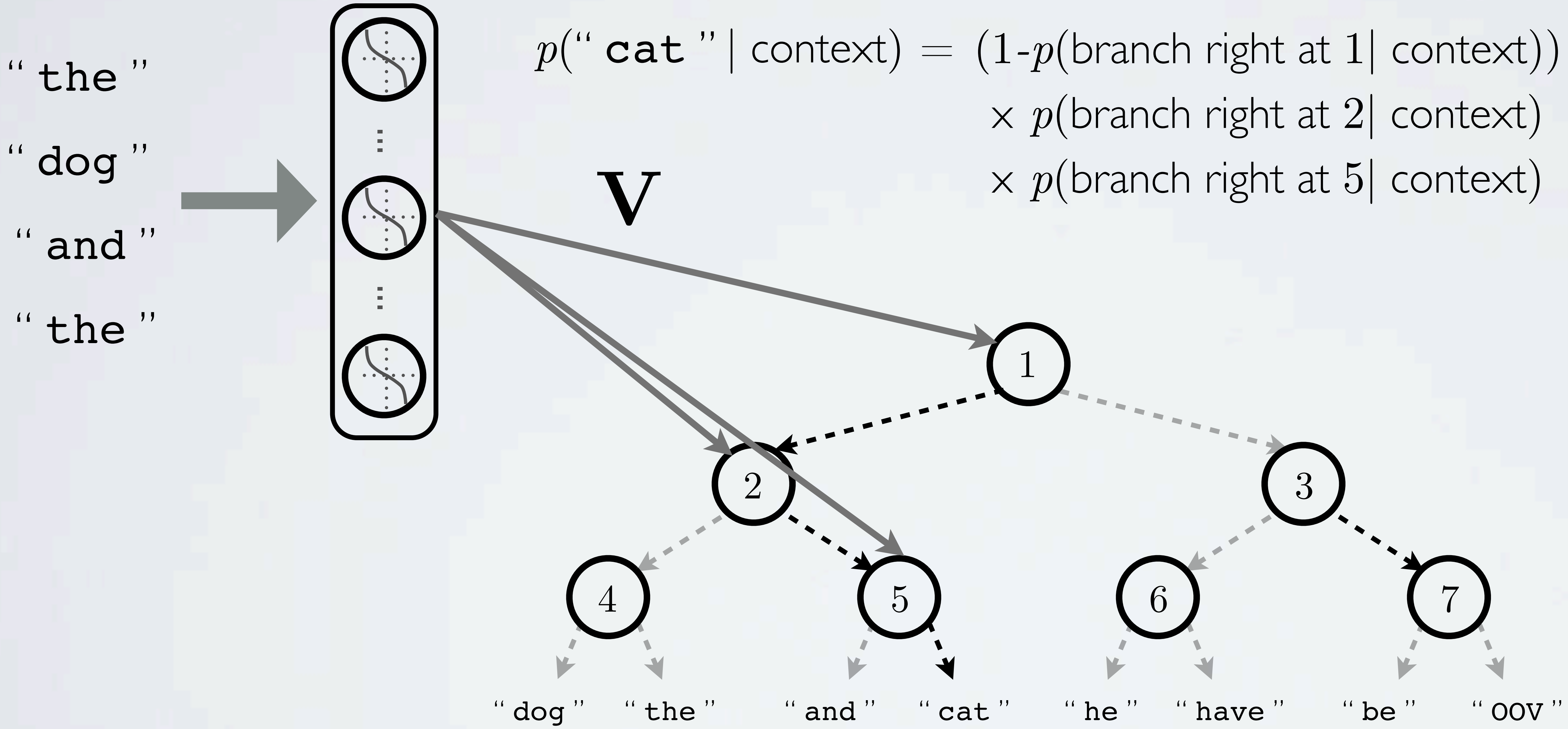


NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

- Example: [" **the** "," **dog** "," **and** ", " **the** ", " **cat** "]

$$p(\text{" }\texttt{cat}\text{ "} \mid \text{context}) = (1\text{-}p(\text{branch right at } 1 \mid \text{context})) \times p(\text{branch right at } 2 \mid \text{context}) \times p(\text{branch right at } 5 \mid \text{context})$$

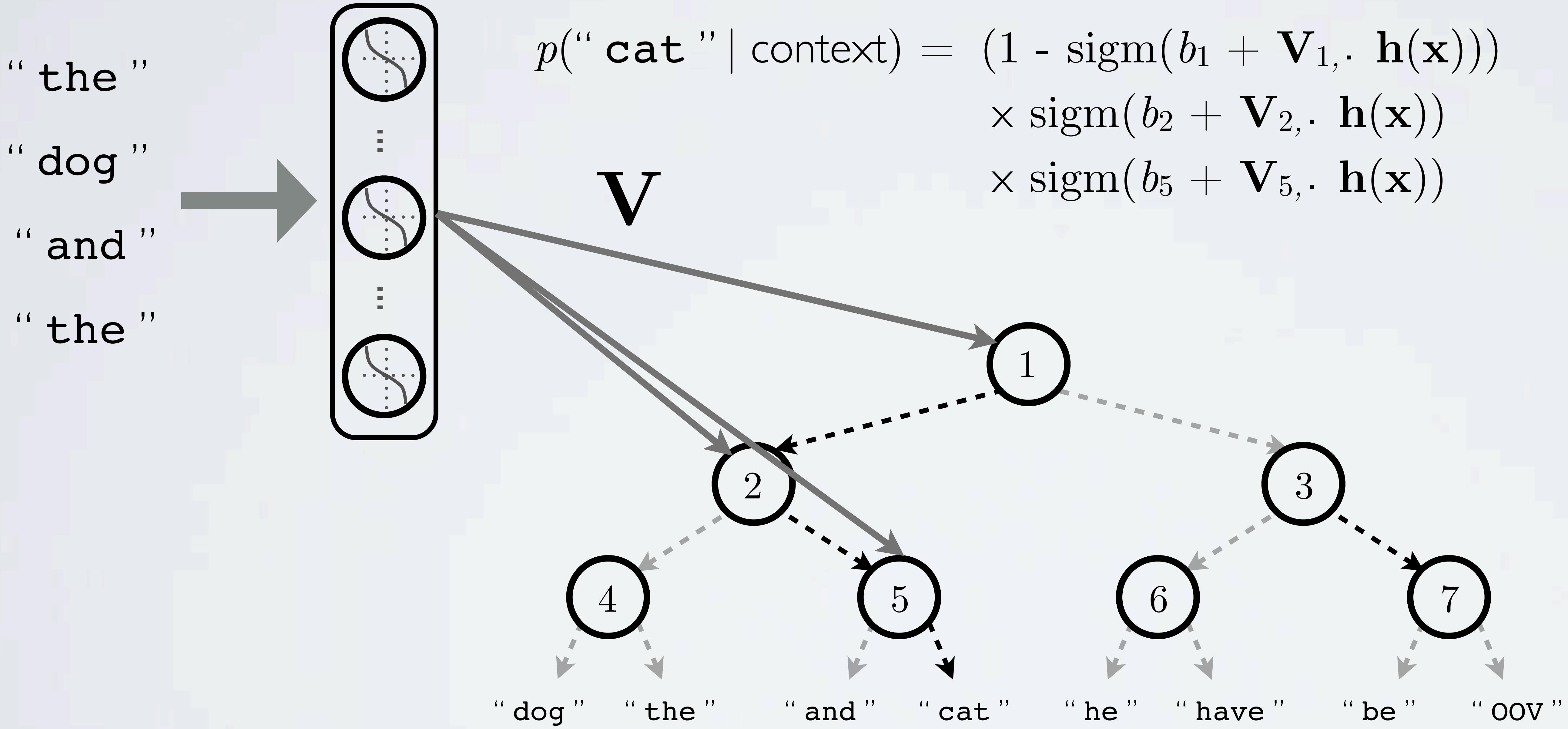


NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

- Example: [" `the` "," `dog` "," `and` ", " `the` ", " `cat` "]

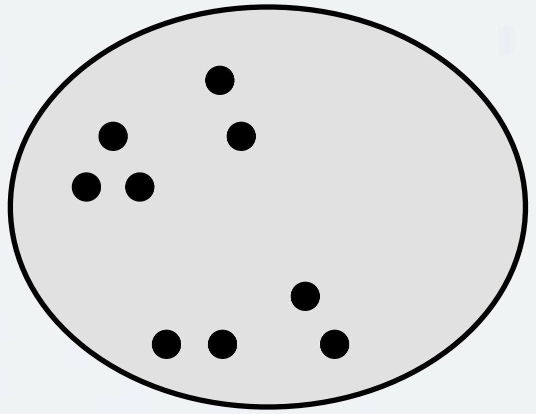
$$p(\text{" }\texttt{cat}\text{ "} \mid \text{context}) = (1 - \text{sigm}(b_1 + \mathbf{V}_{1,\cdot}\, \mathbf{h}(\mathbf{x}))) \times \text{sigm}(b_2 + \mathbf{V}_{2,\cdot}\, \mathbf{h}(\mathbf{x})) \times \text{sigm}(b_5 + \mathbf{V}_{5,\cdot}\, \mathbf{h}(\mathbf{x}))$$



NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

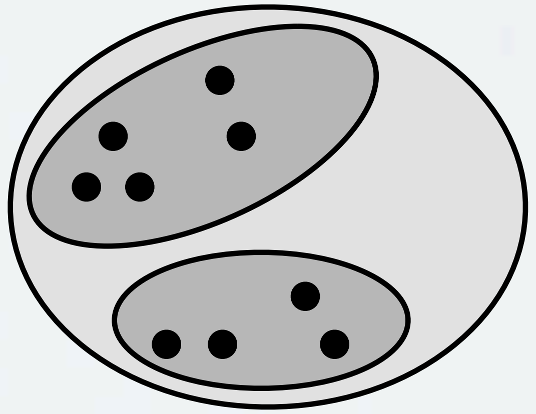
- How to define the word hierarchy
  - can use a randomly generated tree
    - this is likely to be suboptimal
  - can use existing linguistic resources, such as WordNet
    - Hierarchical Probabilistic Neural Network Language Model
      Morin and Bengio, 2005
    - they report a speedup of 258x, with a slight decrease in performance
  - can learn the hierarchy using a recursive partitioning strategy
    - A Scalable Hierarchical Distributed Language Model
      Mnih and Hinton, 2008
    - similar speedup factors are reported, without a performance decrease



NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

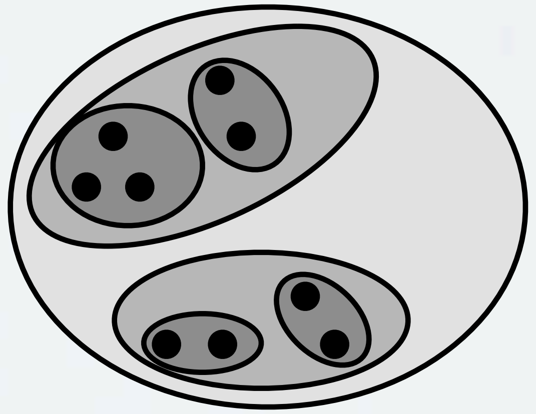
- How to define the word hierarchy
  - can use a randomly generated tree
    - this is likely to be suboptimal
  - can use existing linguistic resources, such as WordNet
    - Hierarchical Probabilistic Neural Network Language Model
      Morin and Bengio, 2005
    - they report a speedup of 258x, with a slight decrease in performance
  - can learn the hierarchy using a recursive partitioning strategy
    - A Scalable Hierarchical Distributed Language Model
      Mnih and Hinton, 2008
    - similar speedup factors are reported, without a performance decrease



NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

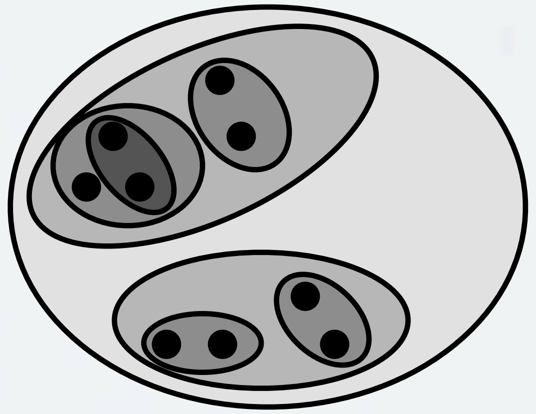
- How to define the word hierarchy
  - can use a randomly generated tree
    - this is likely to be suboptimal
  - can use existing linguistic resources, such as WordNet
    - Hierarchical Probabilistic Neural Network Language Model
      Morin and Bengio, 2005
    - they report a speedup of 258x, with a slight decrease in performance
  - can learn the hierarchy using a recursive partitioning strategy
    - A Scalable Hierarchical Distributed Language Model
      Mnih and Hinton, 2008
    - similar speedup factors are reported, without a performance decrease



NEURAL NETWORK LANGUAGE MODEL

Topics: hierarchical output layer

- How to define the word hierarchy
  - can use a randomly generated tree
    - this is likely to be suboptimal
  - can use existing linguistic resources, such as WordNet
    - Hierarchical Probabilistic Neural Network Language Model
      Morin and Bengio, 2005
    - they report a speedup of 258x, with a slight decrease in performance
  - can learn the hierarchy using a recursive partitioning strategy
    - A Scalable Hierarchical Distributed Language Model
      Mnih and Hinton, 2008
    - similar speedup factors are reported, without a performance decrease



Neural networks

Natural language processing - word tagging

WORD TAGGING

Topics: word tagging

- In many NLP applications, it is useful to augment text data with syntactic and semantic information
  - we would like to add syntactic/semantic labels to each word
- This problem can be tackled using a conditional random field with neural network unary potentials
  - we will describe the model developed by Ronan Collobert and Jason Weston in:

    A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning
    Collobert and Weston, 2008

    (see Natural Language Processing (Almost) from Scratch for the journal version)

WORD TAGGING

Topics: part-of-speech tagging

- Tag each word with its part of speech category
  - noun, verb, adverb, etc.
  - might want to distinguish between singular/plural, present tense/past tense, etc.
  - see Penn Treebank POS tags set for an example

- Example:

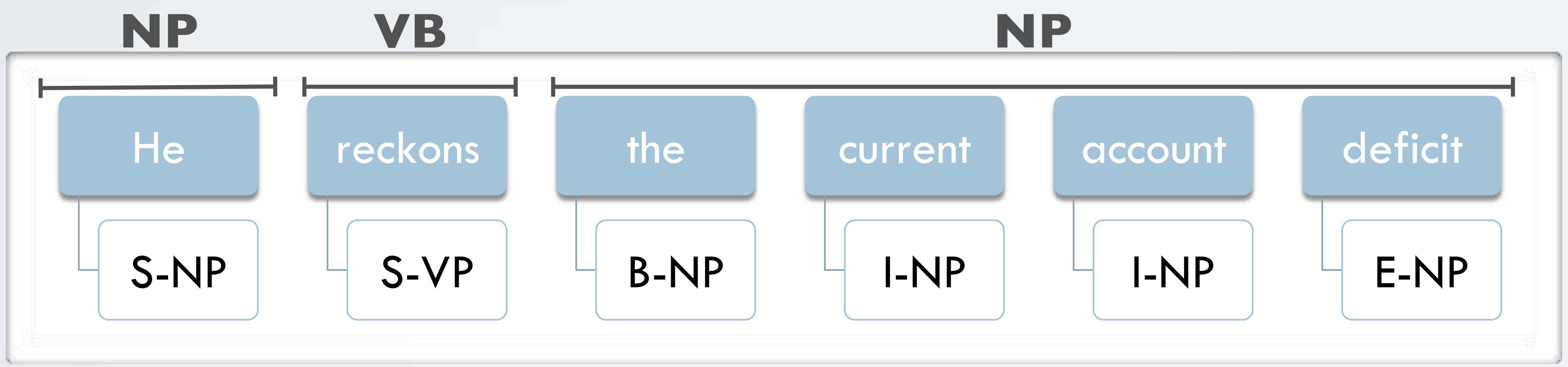
The	little	yellow	dog	barked	at	the	cat
DT	JJ	JJ	NN	VBD	IN	DT	NN

(from Stanislas Lauly)

WORD TAGGING

Topics: chunking

- Segment phrases into syntactic phrases
  - noun phrase, verb phrase, etc.
- Segments are identified with IOBES encoding
  - single word phrase (`S-` prefix). Ex.: `S-NP`
  - multiword phrase (`B-`, `I-`, `E-` prefixes). Ex.: `B-VP I-VP I-VP E-VP`
  - words outside of syntactic phrases: `O`



(from Stanislas Lauly)

WORD TAGGING

Topics: named entity recognition (NER)

- Identify phrases referring to a named entity
  - person
  - location
  - organization

- Example:

U.N.	official	Ekeus	heads	for	Baghdad
S-ORG	O	S-PER	O	O	S-LOC

(from Stanislas Lauly)

WORD TAGGING

Topics: semantic role labeling (SRL)

- For each verb, identify the role of other words with respect to that verb
- Example:
  - `V`: verb
  - `A0`: acceptor
  - `A1`: thing accepted
  - `A2`: accepted from
  - `A3`: attribute
  - `AM-MOD`: modal
  - `AM-NEG`: negation

He	would	n’t	accept	anything	of	value
S-A0	S-AM-MOD	S-AM-NEG	V	B-A1	I-A1	E-A1

(from Stanislas Lauly)

WORD TAGGING

Topics: labeled corpus

- The raw data looks like this:

The	DT	B-NP	O	B-A0	B-A0
$	$	I-NP	O	I-A0	I-A0
1.4	CD	I-NP	O	I-A0	I-A0
billion	CD	I-NP	O	I-A0	I-A0
robot	NN	I-NP	O	I-A0	I-A0
spacecraft	NN	E-NP	O	E-A0	E-A0
faces	VBZ	S-VP	O	S-V	O
a	DT	B-NP	O	B-A1	O
six-year	JJ	I-NP	O	I-A1	O
journey	NN	E-NP	O	I-A1	O
to	TO	B-VP	O	I-A1	O
explore	VB	E-VP	O	I-A1	S-V
Jupiter	NNP	S-NP	S-ORG	I-A1	B-A1
and	CC	O	O	I-A1	I-A1
its	PRP$	B-NP	O	I-A1	I-A1
16	CD	I-NP	O	I-A1	I-A1
known	JJ	I-NP	O	I-A1	I-A1
moons	NNS	E-NP	O	E-A1	E-A1
.	.	O	O	O	O

Neural networks

Natural language processing - convolutional network

WORD TAGGING

Topics: word tagging

- In many NLP applications, it is useful to augment text data with syntactic and semantic information
  - we would like to add syntactic/semantic labels to each word
- This problem can be tackled using a conditional random field with neural network unary potentials
  - we will describe the model developed by Ronan Collobert and Jason Weston in:

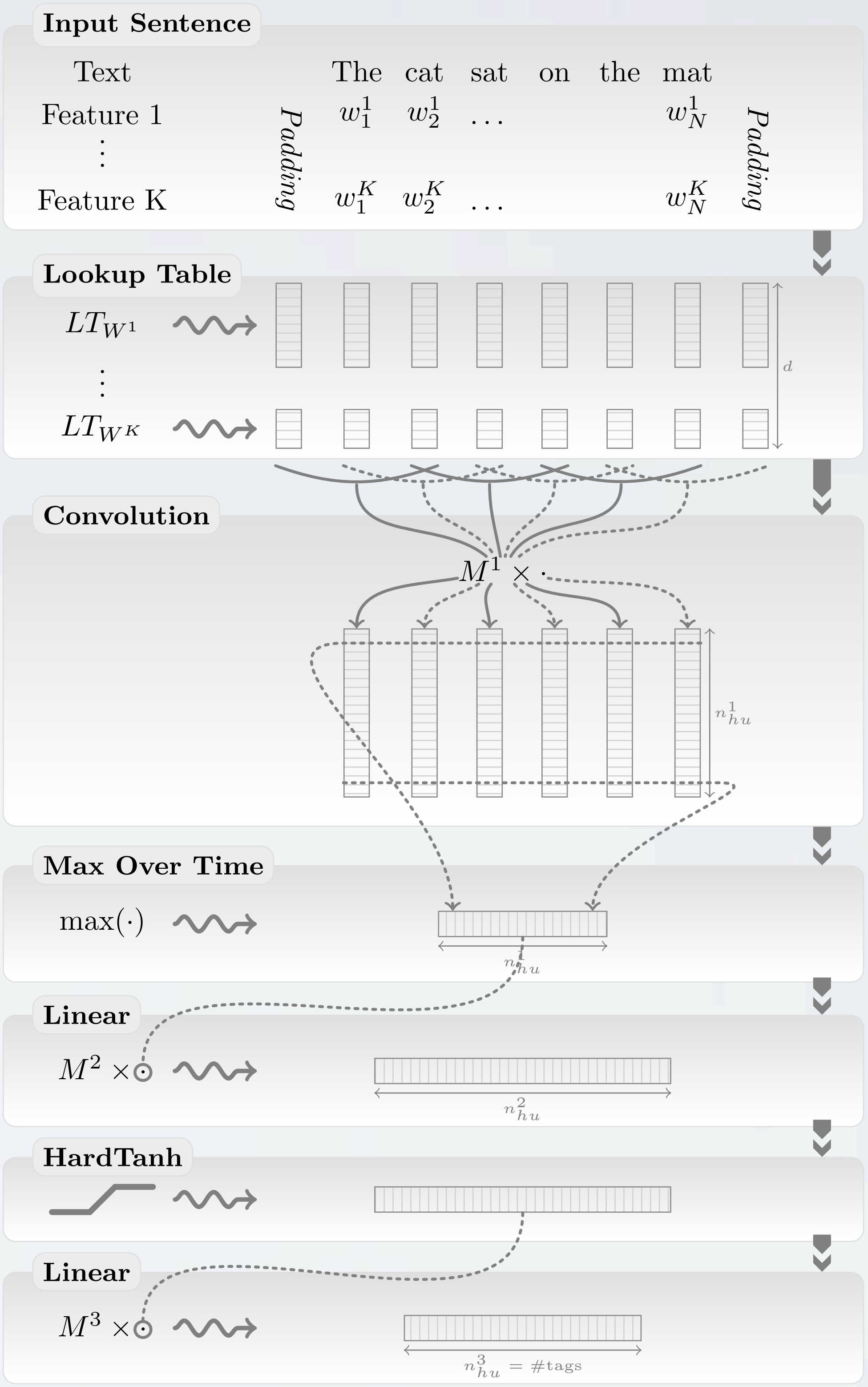
    A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning
    Collobert and Weston, 2008

    (see Natural Language Processing (Almost) from Scratch for the journal version)

SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

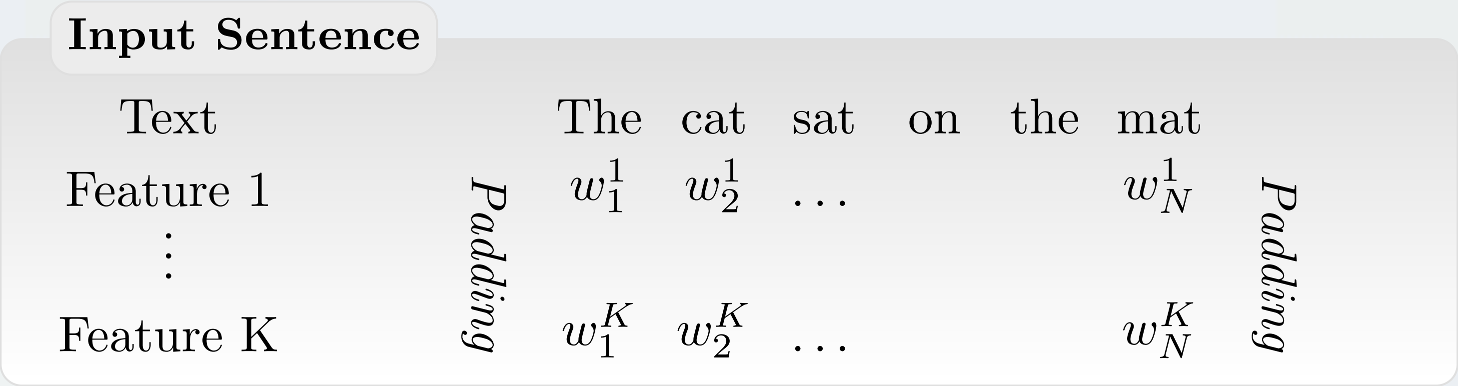
- How to model each label sequence
  - could use a CRF with neural network unary potentials, based on a window (context) of words
    - not appropriate for semantic role labeling, because relevant context might be very far away
  - Collobert and Weston suggest a convolutional network over the whole sentence
    - prediction at a given position can exploit information from any word in the sentence



SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

- Each word can be represented by more than one feature
  - feature of the word itself
  - substring features
    - prefix: " `eating` " → " `eat` "
    - suffix: " `eating` " → " `ing` "
  - gazetteer features
    - whether the word belong to a list of known locations, persons, etc.
- These features are treated like word features, with their own lookup tables



SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

- Feature must encode for which word we are making a prediction
  - done by adding the relative position $i\text{-}pos_w$, where pos_w is the position of the current word
  - this feature also has its lookup table

Input Sentence

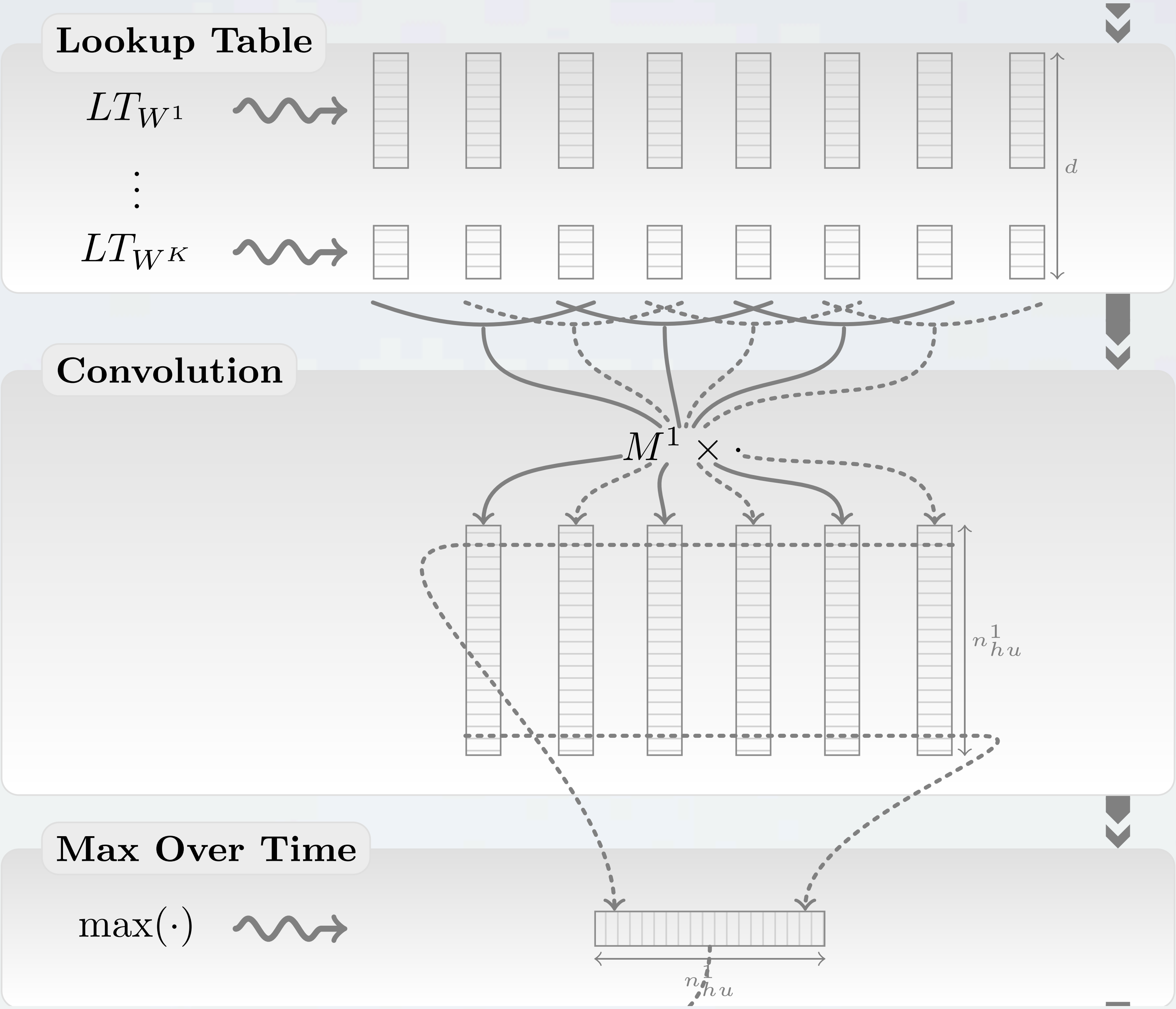
Text	*Padding*	The	cat	sat	on	the	mat	*Padding*
Feature 1		w_1^1	w_2^1	$\ldots$			w_N^1	
$\vdots$								
Feature K		w_1^K	w_2^K	$\ldots$			w_N^K	

- For SRL, must know the roles for which verb we are predicting
  - also add the relative position of that verb $i\text{-}pos_v$

SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

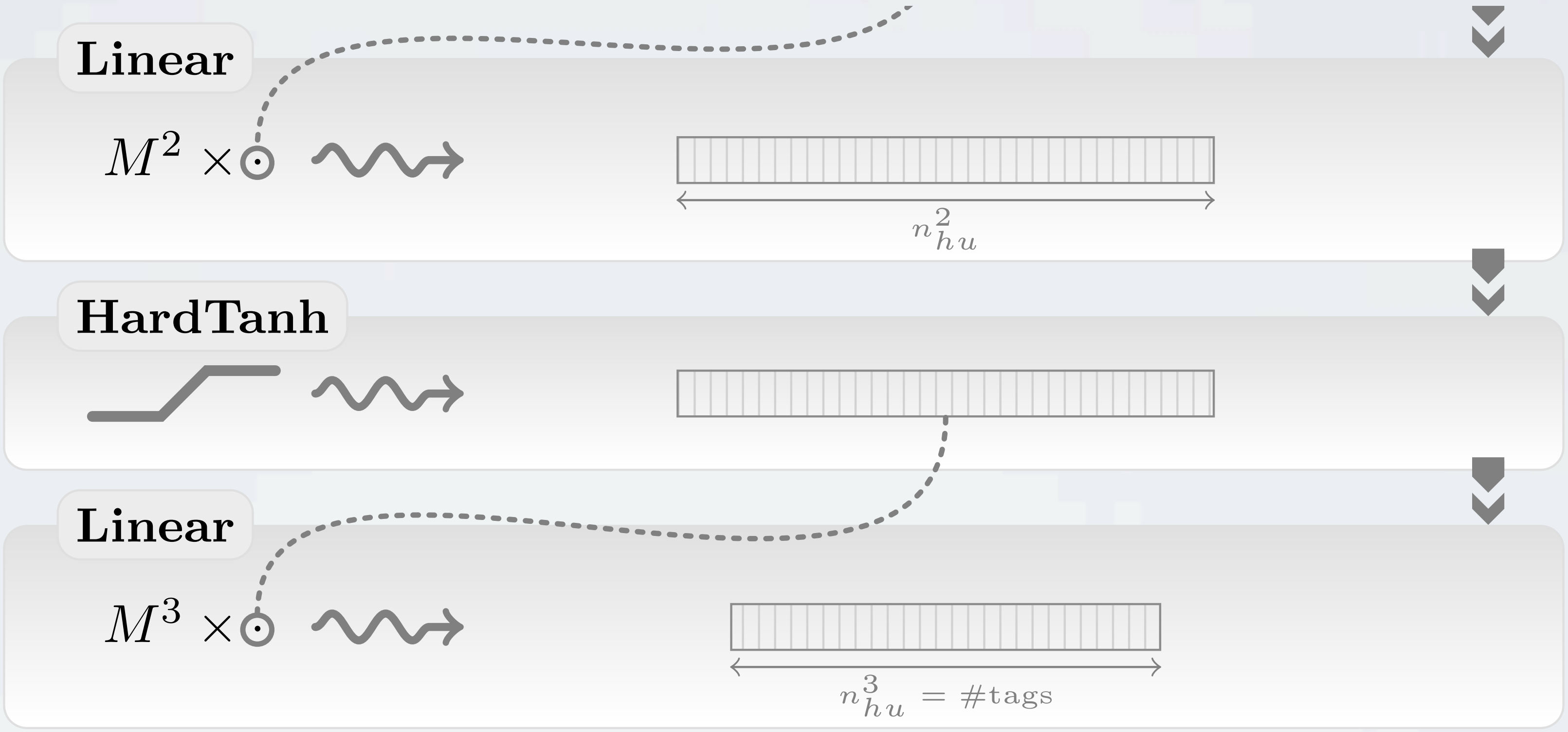
- Lookup table:
  - for each word concatenate the representations of its features
- Convolution:
  - at every position, compute linear activations from a window of representations
  - this is a convolution in 1D
- Max pooling:
  - obtain a fixed hidden layer with a max across positions



SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

- Regular neural network:
  - the pooled representation serves as the input of a regular neural network
  - they proposed using a "hard" version of the tanh activation function



- The outputs are used as the unary potential of a chain CRF over the labels
  - no connections between the CRFs of the different task (one CRF per task)
  - a separate neural network is used for each task

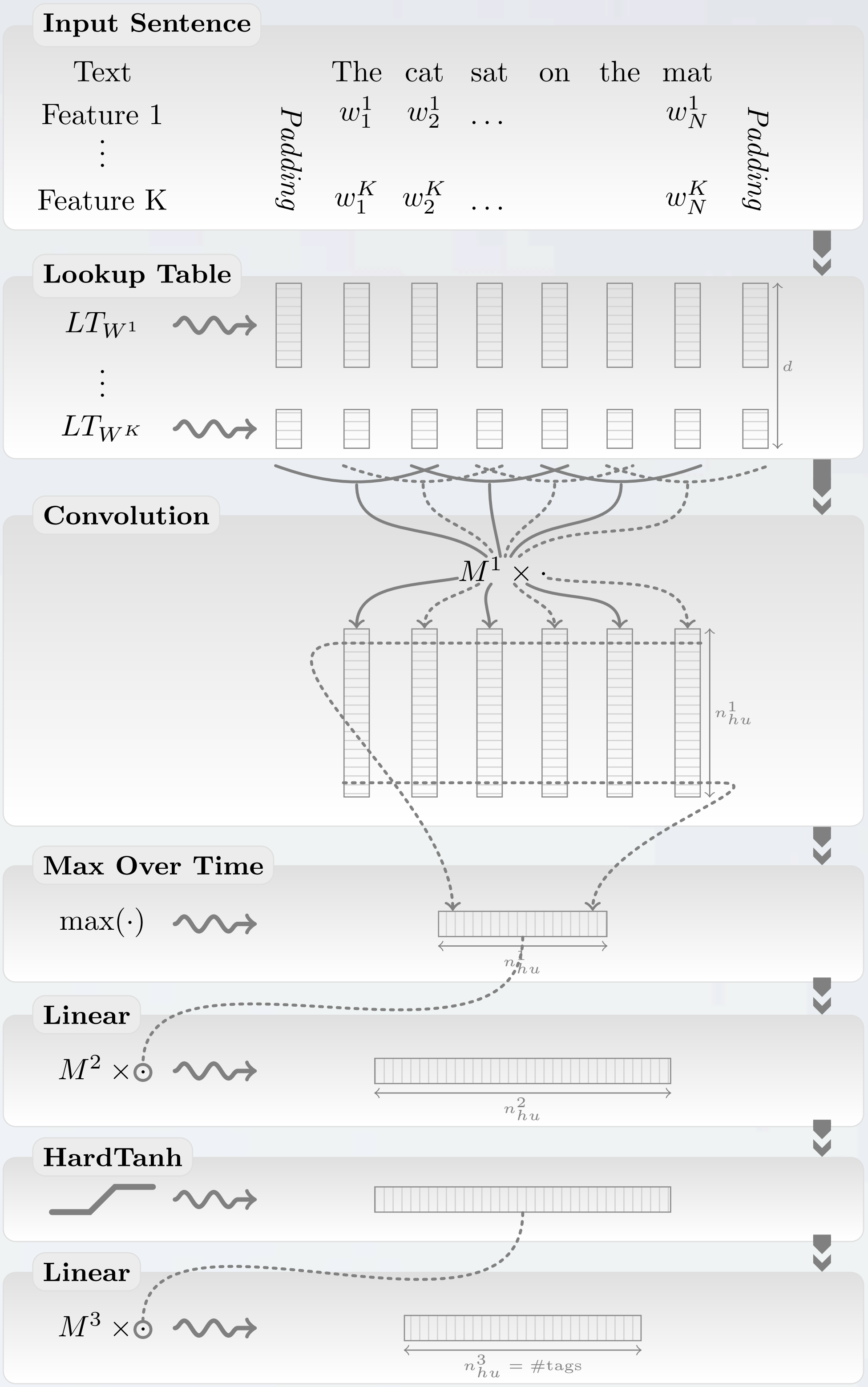
Neural networks

Natural language processing - multitask learning

SENTENCE NEURAL NETWORK

Topics: sentence convolutional network

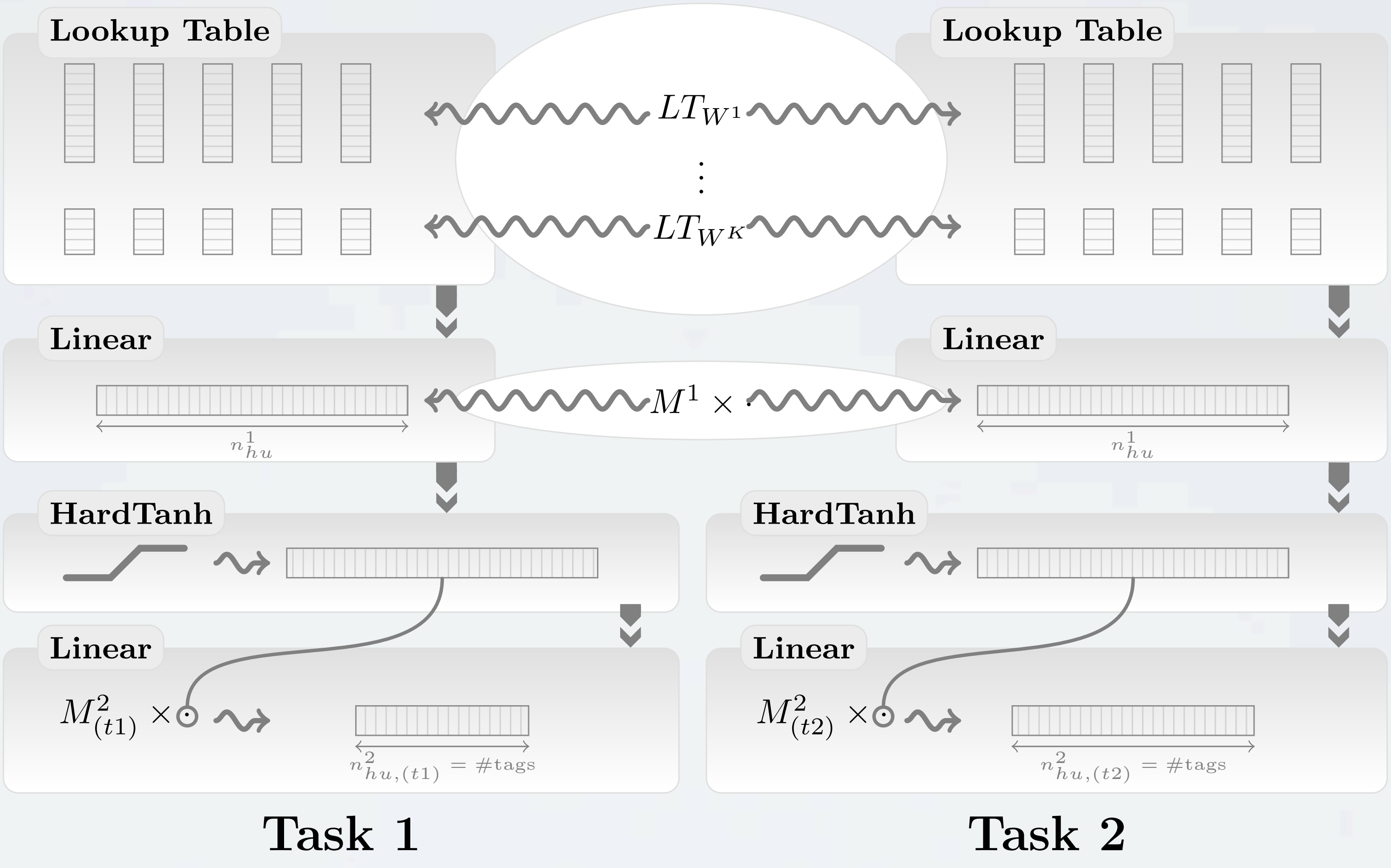
- How to model each label sequence
  - could use a CRF with neural network unary potentials, based on a window (context) of words
    - not appropriate for semantic role labeling, because relevant context might be very far away
  - Collobert and Weston suggest a convolutional network over the whole sentence
    - prediction at a given position can exploit information from any word in the sentence



SENTENCE NEURAL NETWORK

Topics: multitask learning

- Could share vector representations of the features across tasks
  - simply use the same lookup tables across tasks
  - the other parameters of the neural networks are not tied



- This is referred to as multitask learning
  - the idea is to transfer knowledge learned within the word representations across the different task

SENTENCE NEURAL NETWORK

Topics: language model

- We can design other tasks without any labeled data
  - identify whether the middle word of a window of text is an "impostor"

    "`cat sat` **`on`** `the mat`" vs "`cat sat` **`think`** `the mat`"
  - can generate impostor examples from unlabeled text (Wikipedia)
    - pick a window of words from unlabeled corpus
    - replace middle word with a different, randomly chosen word
  - train a neural network (with word representations) to assign a higher score to the original window

    original window

    $$\max\left\{0\,,\,1 - f_\theta(x) + f_\theta(x^{(w)})\right\}$$

    impostor window with word w
  - similar to language modeling, except we predict the word in the middle

SENTENCE NEURAL NETWORK

Topics: experimental comparison

- From Natural Language Processing (Almost) from Scratch by Collobert et al.

Approach	POS (PWA)	CHUNK (F1)	NER (F1)	SRL (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+SLL	96.37	90.33	81.47	70.99

SENTENCE NEURAL NETWORK

Topics: experimental comparison

- From Natural Language Processing (Almost) from Scratch by Collobert et al.

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+SLL	96.37	90.33	81.47	70.99
NN+SLL+LM2	97.12	93.37	88.78	74.15
NN+SLL+LM2+MTL	97.22	93.75	88.27	74.29

SENTENCE NEURAL NETWORK

Topics: experimental comparison

- From Natural Language Processing (Almost) from Scratch by Collobert et al.

Approach	POS (PWA)	CHUNK (F1)	NER (F1)	SRL (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+SLL	96.37	90.33	81.47	70.99
NN+SLL+LM2	97.12	93.37	88.78	74.15
NN+SLL+LM2+MTL	97.22	93.75	88.27	74.29
NN+SLL+LM2+Suffix2	97.29	–	–	–
NN+SLL+LM2+Gazetteer	–	–	89.59	–
NN+SLL+LM2+POS	–	94.32	88.67	–
NN+SLL+LM2+CHUNK	–	–	–	74.72

SENTENCE NEURAL NETWORK

Topics: experimental comparison

- Nearest neighbors in word representation space:

FRANCE 454	JESUS 1973	XBOX 6909	REDDISH 11724	SCRATCHED 29869	MEGABITS 87025
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

- For a 2D visualization: http://www.cs.toronto.edu/~hinton/turian.png

Neural networks

Natural language processing - recursive network

FROM WORDS TO PHRASES

Topics: word phrase representations

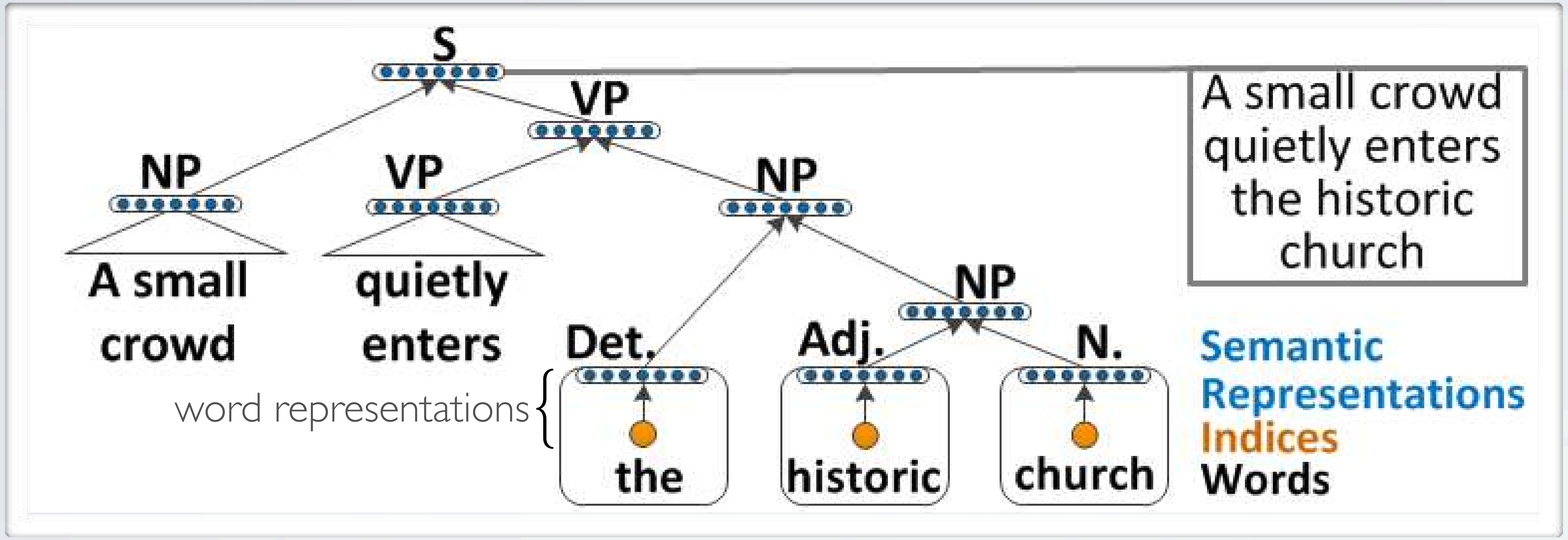
- We've seen how to learn representations for single words

- How could we learn representations for phrases of arbitrary length?
  - can we model relationships between words and multiword expressions
    - ex.: " `consider` " ≈ " `take into account` "
  - can we extract a representation of full sentences that preserves some of its semantic meaning
    - ex.: " `word representations were learned from a corpus` " ≈ " `we trained word representations on a text data set` "

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Idea: recursively merge pairs of word/phrase representations



Socher, Lin, Ng and Manning, 2011

- We need 2 things
  - a model that merges pairs of representations
  - a model that determines the tree structure

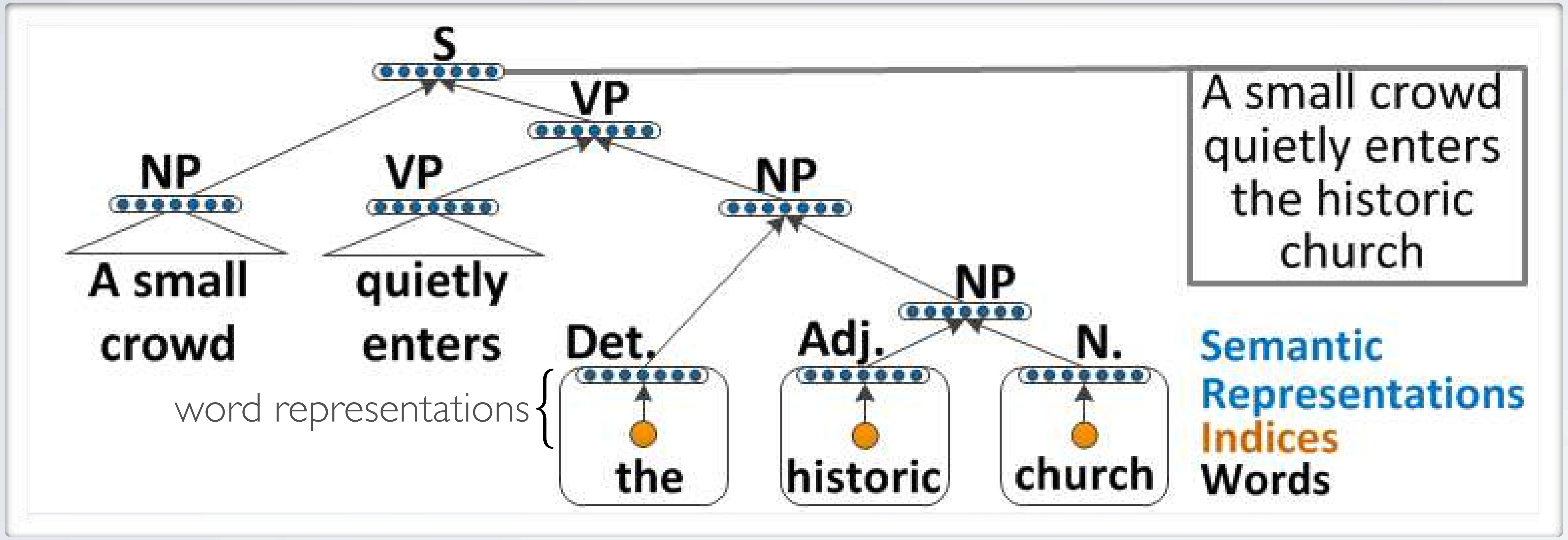
Neural networks

Natural language processing - merging representations

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Idea: recursively merge pairs of word/phrase representations



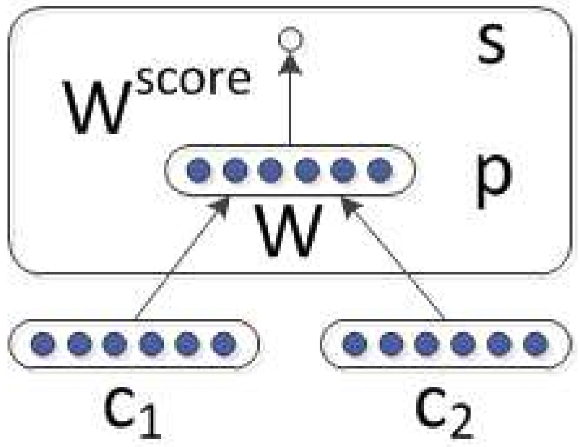
- We need 2 things
  - a model that merges pairs of representations
  - a model that determines the tree structure

Socher, Lin, Ng and Manning, 2011

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Given two input representations c_1 and c_2, the recursive network computes the merged representation p as follows:



$$\begin{aligned} s &= W^{score} p \\ p &= f(W[c_1; c_2] + b) \end{aligned}$$

Socher, Lin,
Ng and Manning, 2011

- The network also computes a score s
  - it estimates the quality of the merge
  - it will be used to decide which pairs of representations to merge first

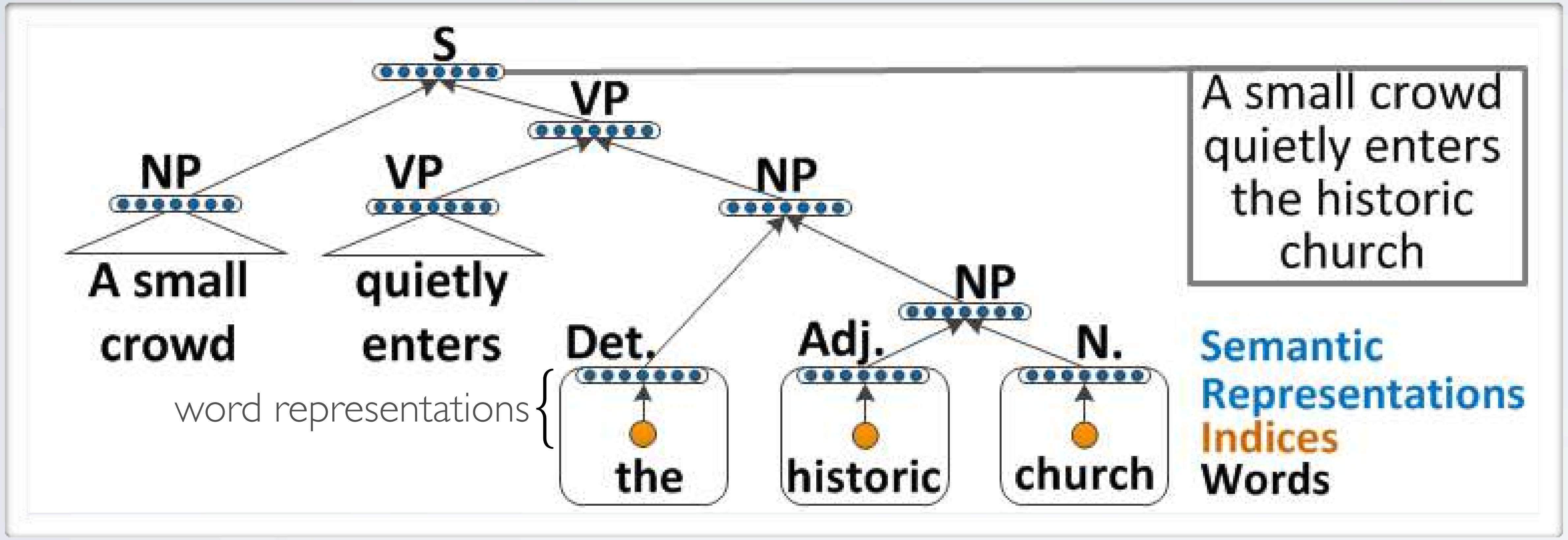
Neural networks

Natural language processing - tree inference

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Idea: recursively merge pairs of word/phrase representations



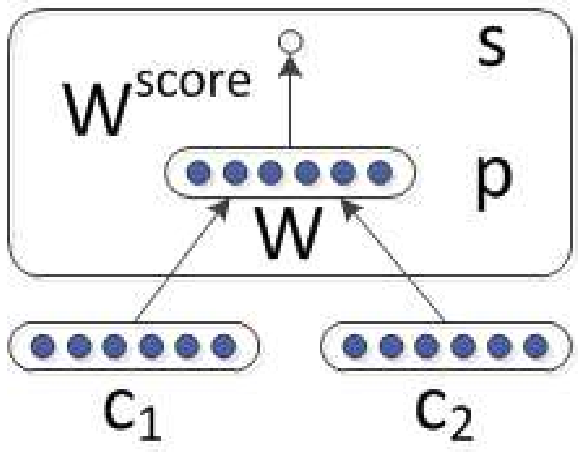
Socher, Lin, Ng and Manning, 2011

- We need 2 things
  - a model that merges pairs of representations
  - a model that determines the tree structure

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Given two input representations c_1 and c_2, the recursive network computes the merged representation p as follows:



$$
\begin{aligned}
s &= W^{score} p \\
p &= f(W[c_1; c_2] + b)
\end{aligned}
$$

Socher, Lin, Ng and Manning, 2011

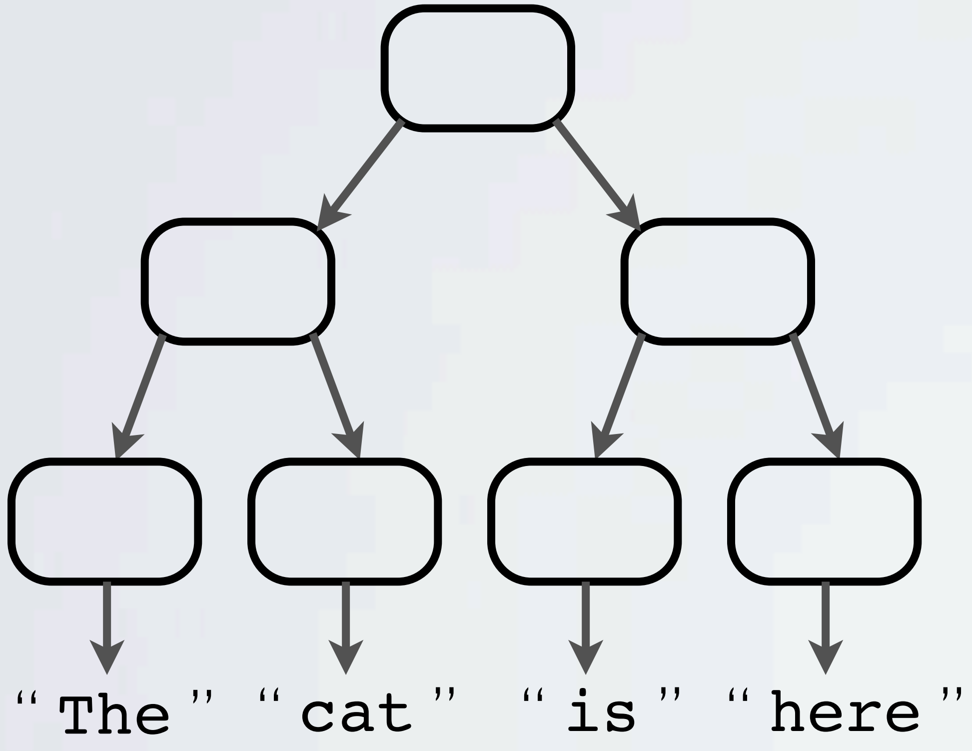
- The network also computes a score s
  - it estimates the quality of the merge
  - it will be used to decide which pairs of representations to merge first

RECURSIVE NEURAL NETWORK

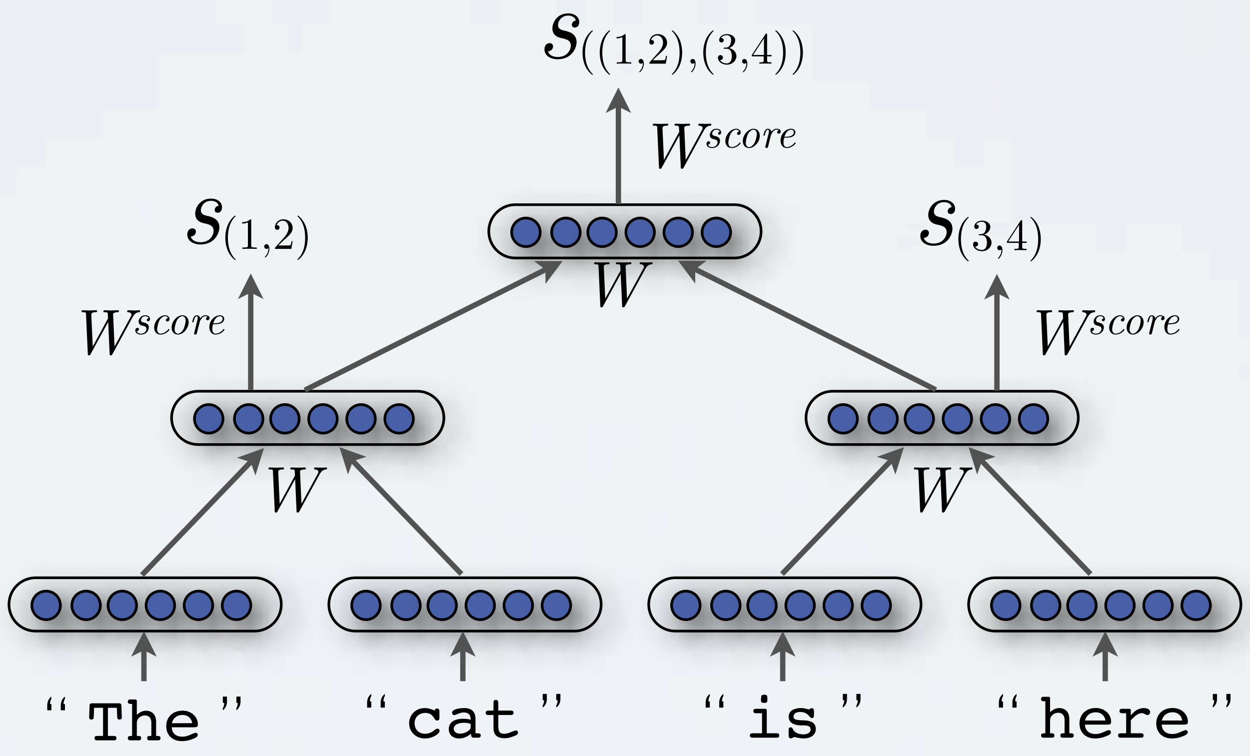
Topics: recursive neural network (RNN)

- The score of the full tree is the sum of all merging scores

Parse tree



Recursive network

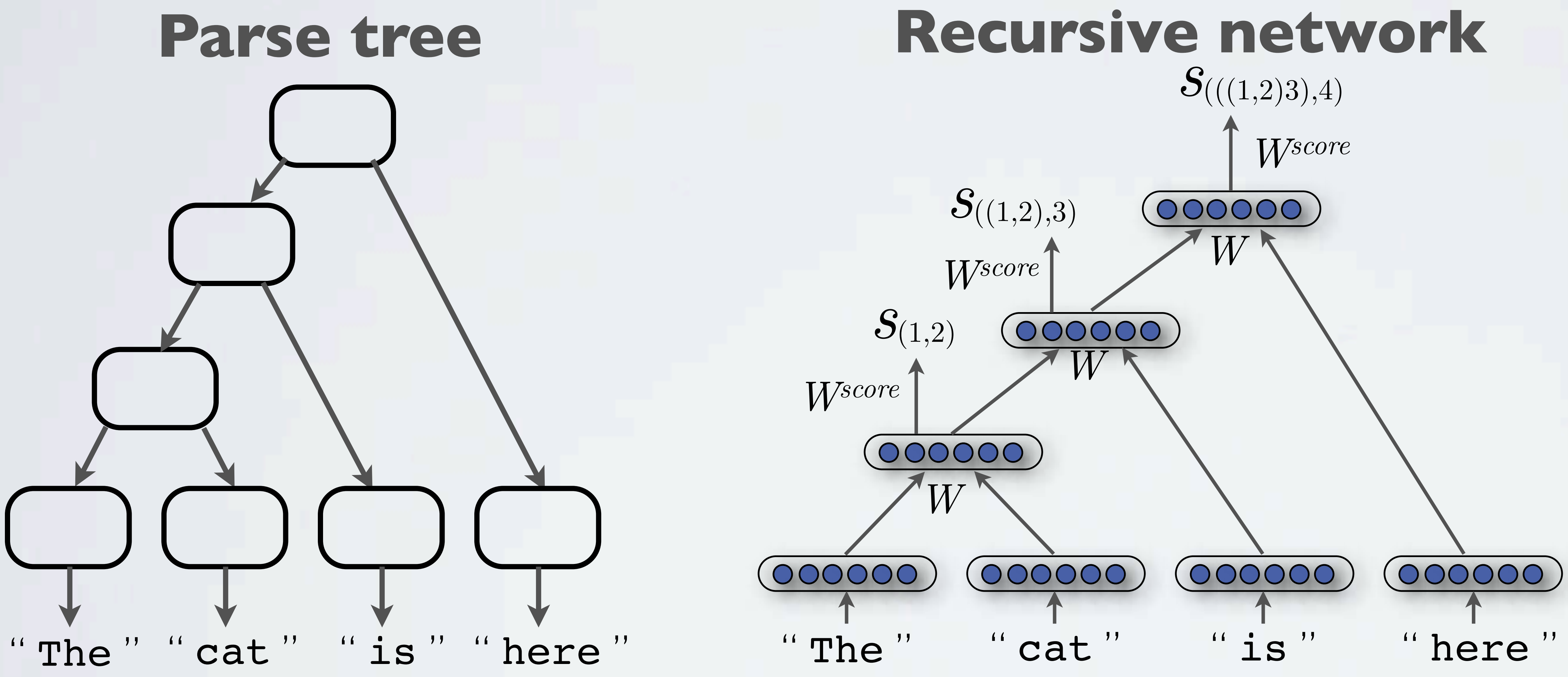


Score: $s_{(1,2)} + s_{(3,4)} + s_{((1,2),(3,4))}$

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- The score of the full tree is the sum of all merging scores

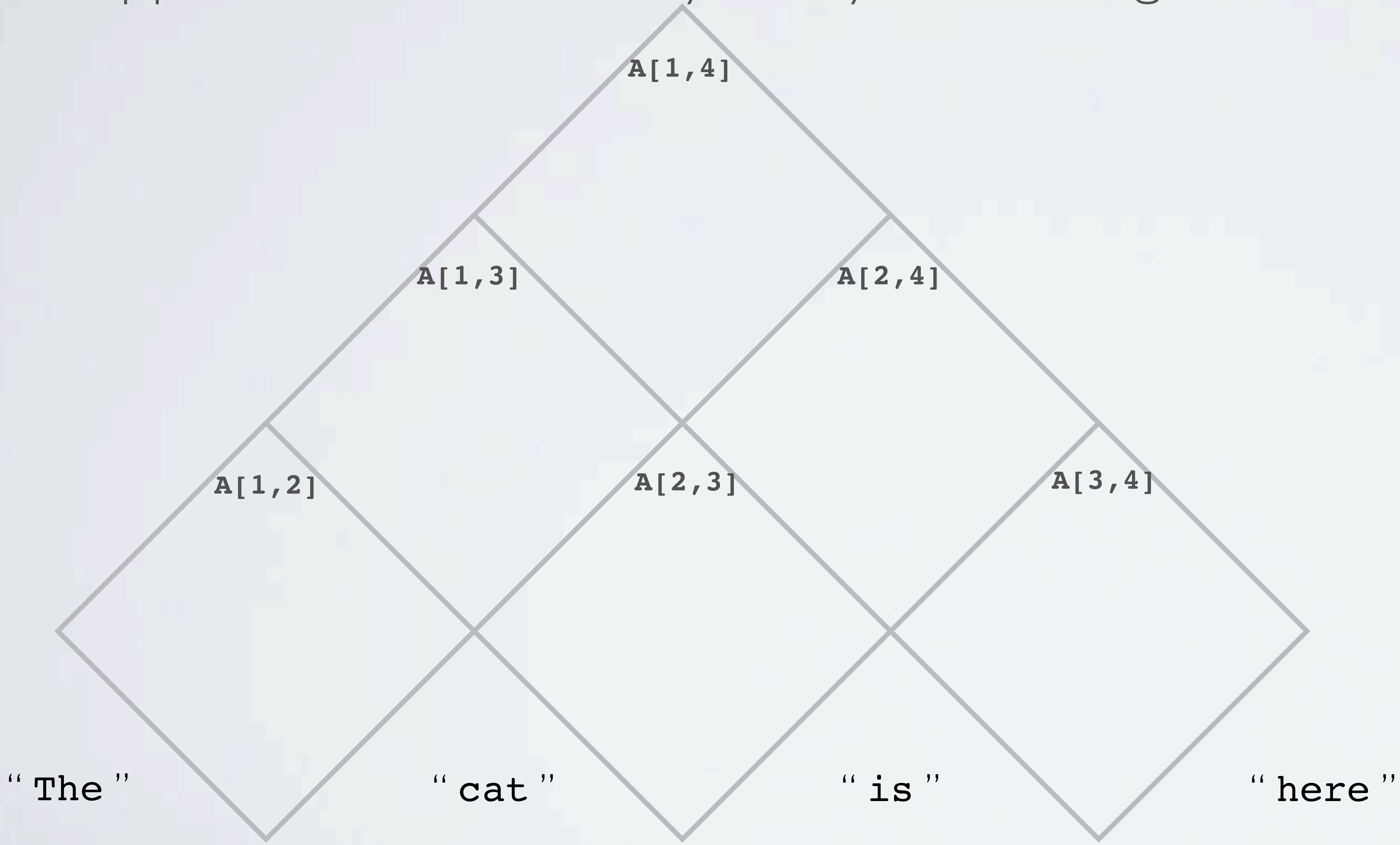


Score: $s_{(1,2)} + s_{((1,2),3)} + s_{(((1,2),3),4)}$

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

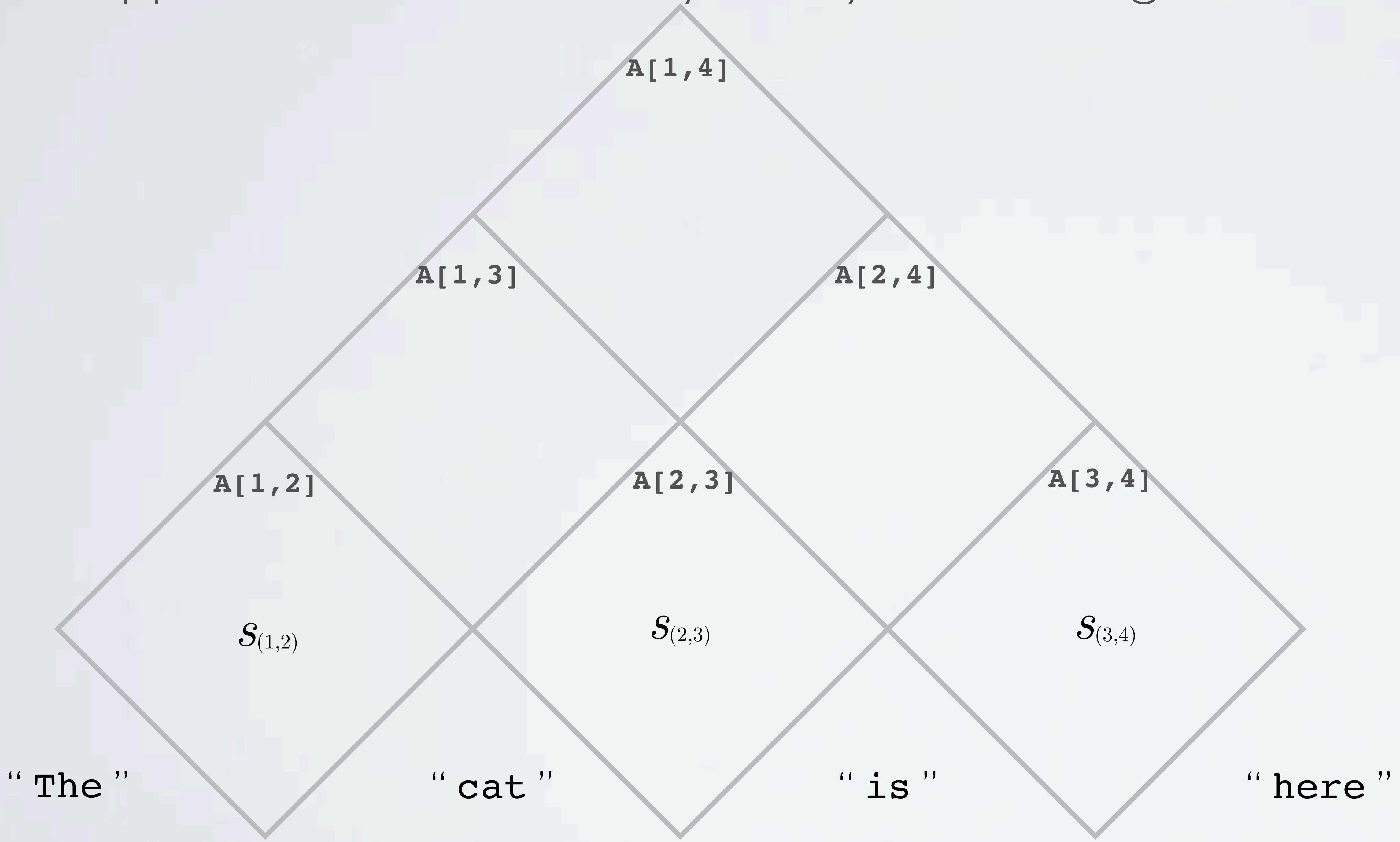
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

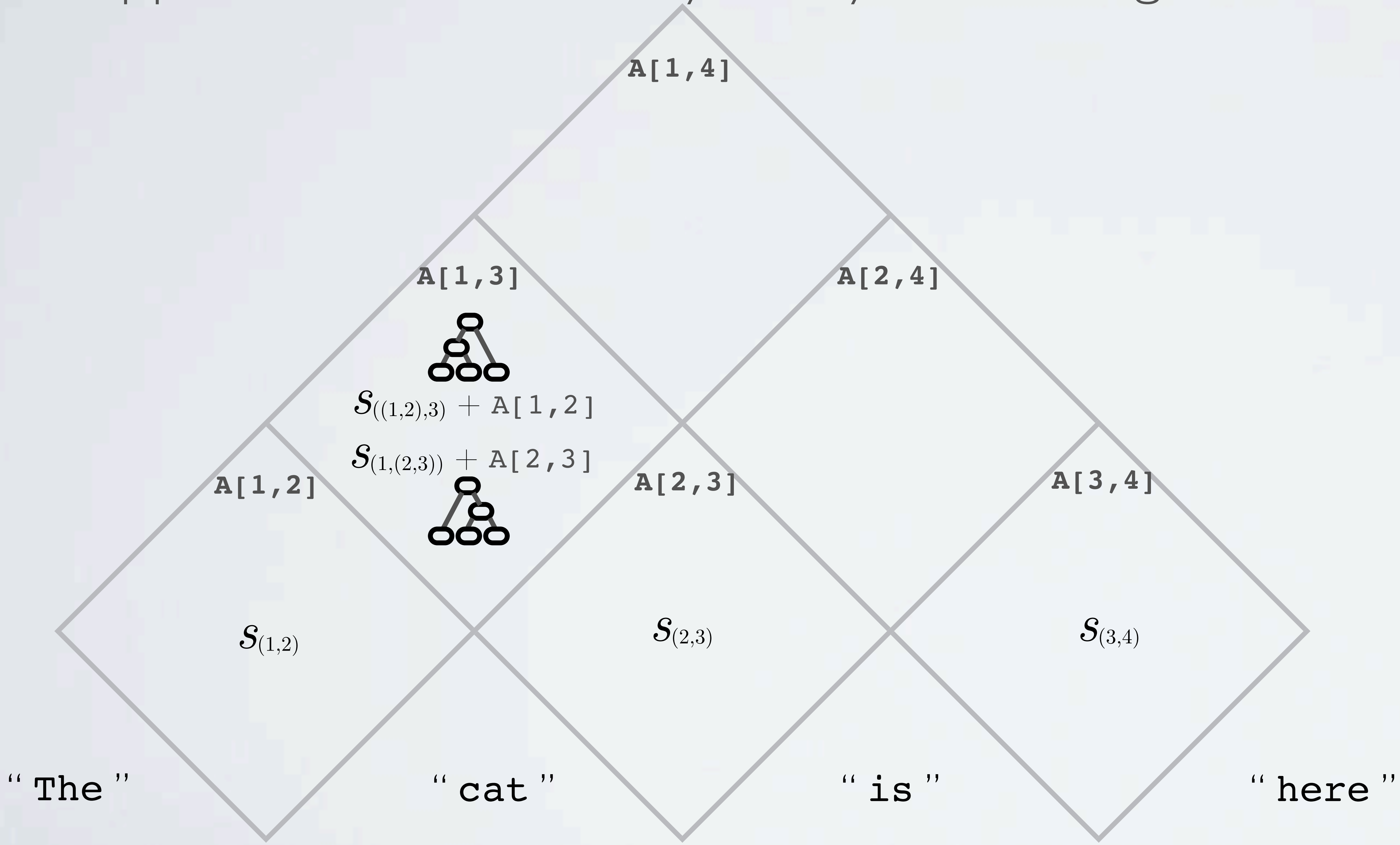
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

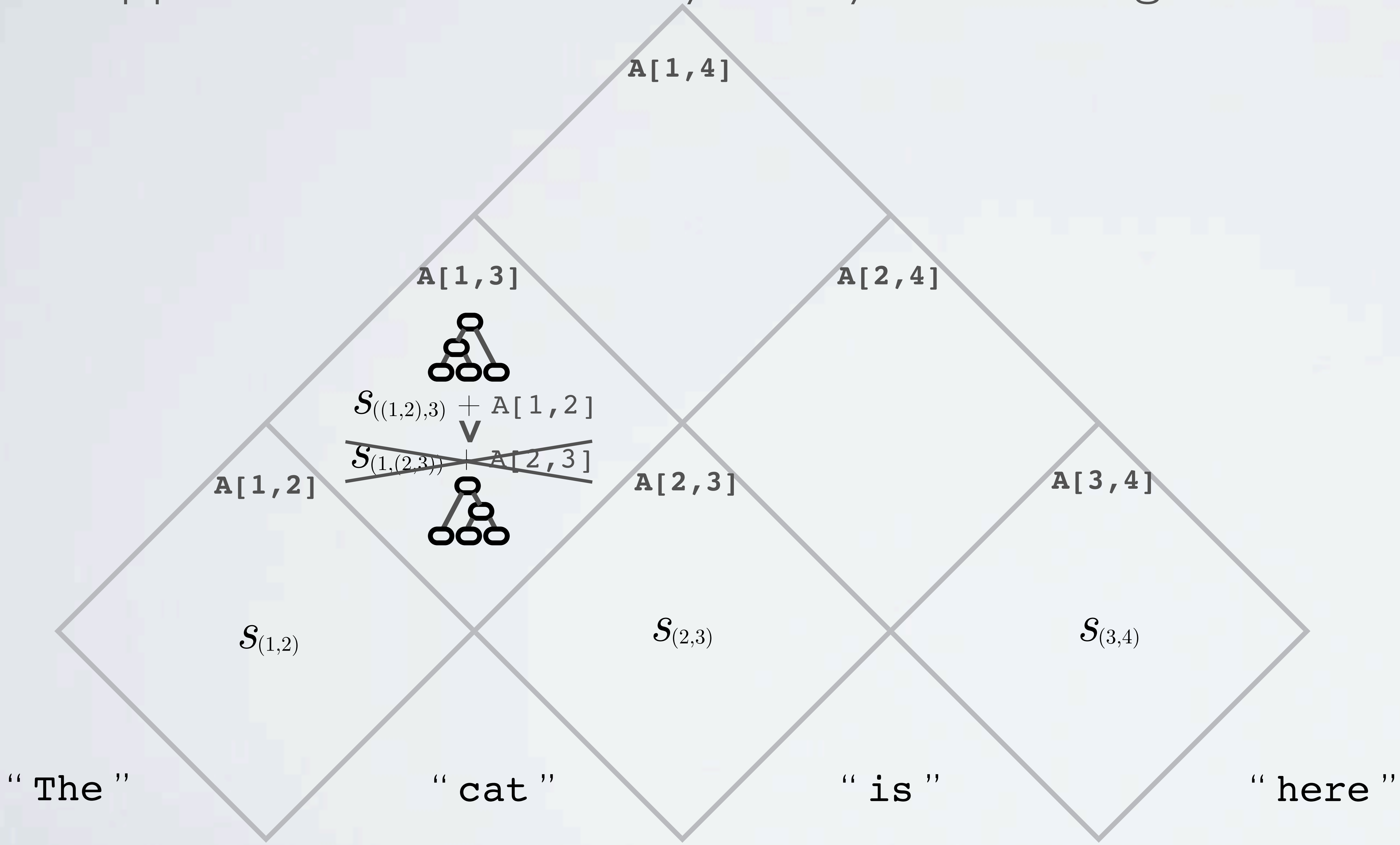
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

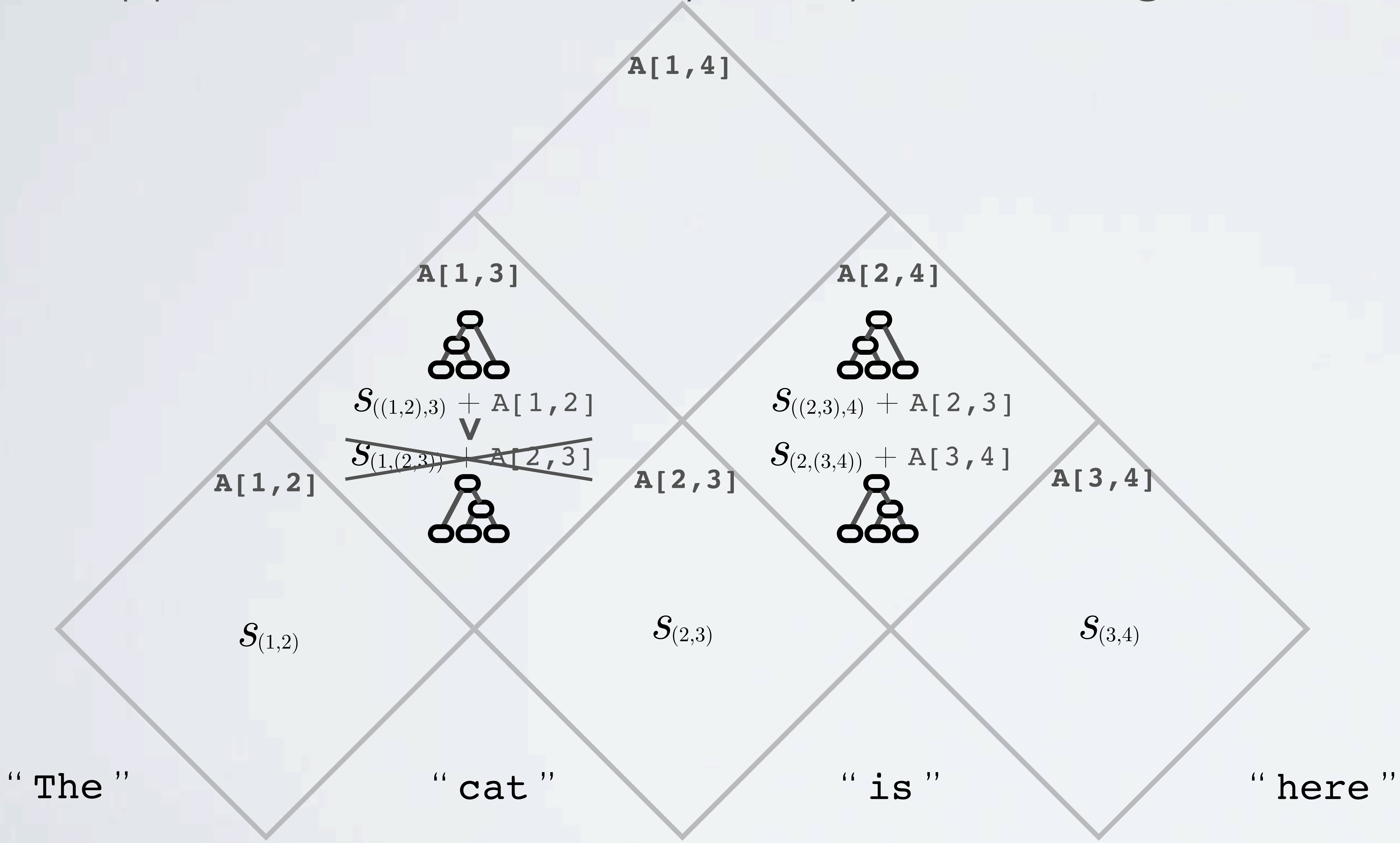
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

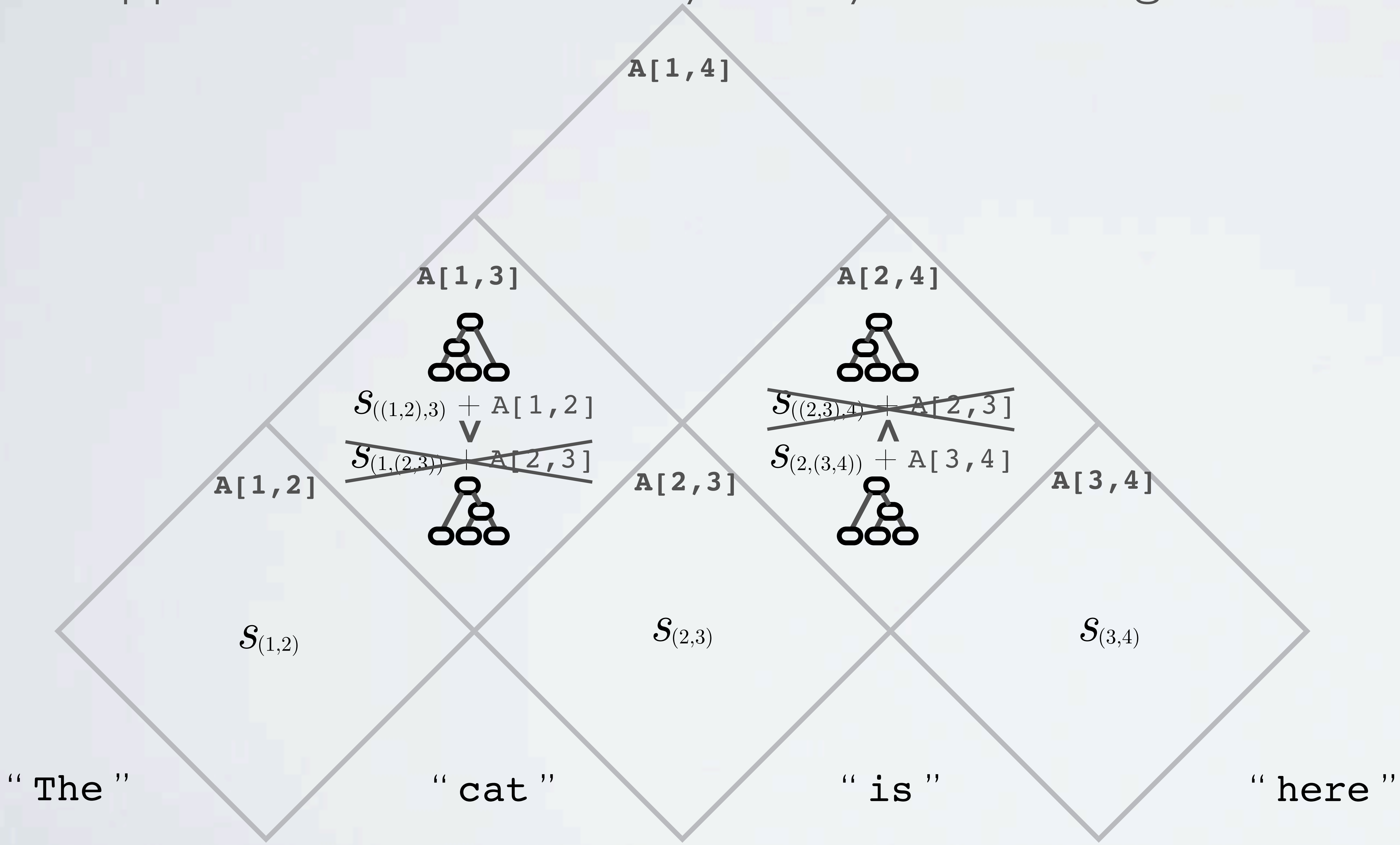
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

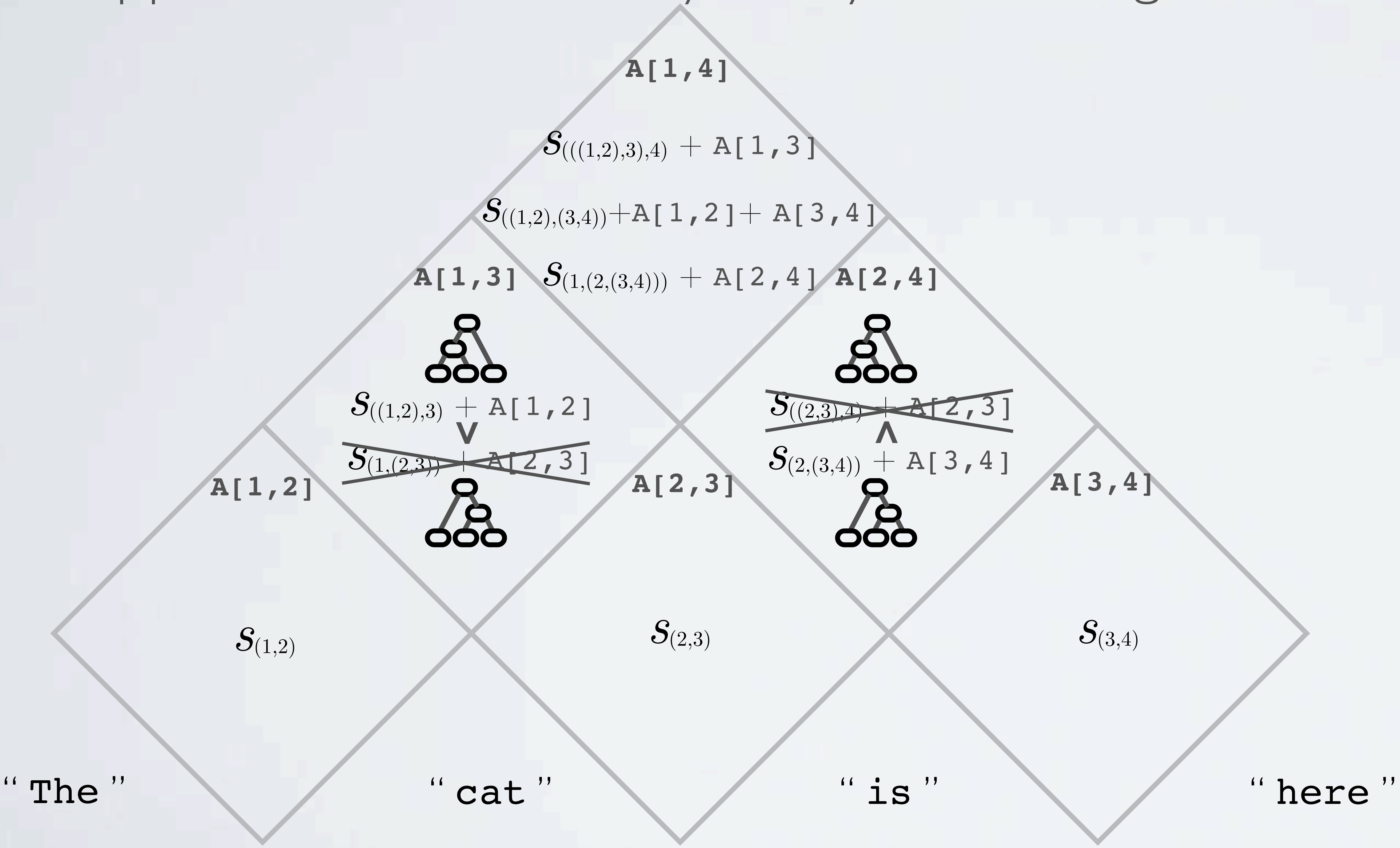
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

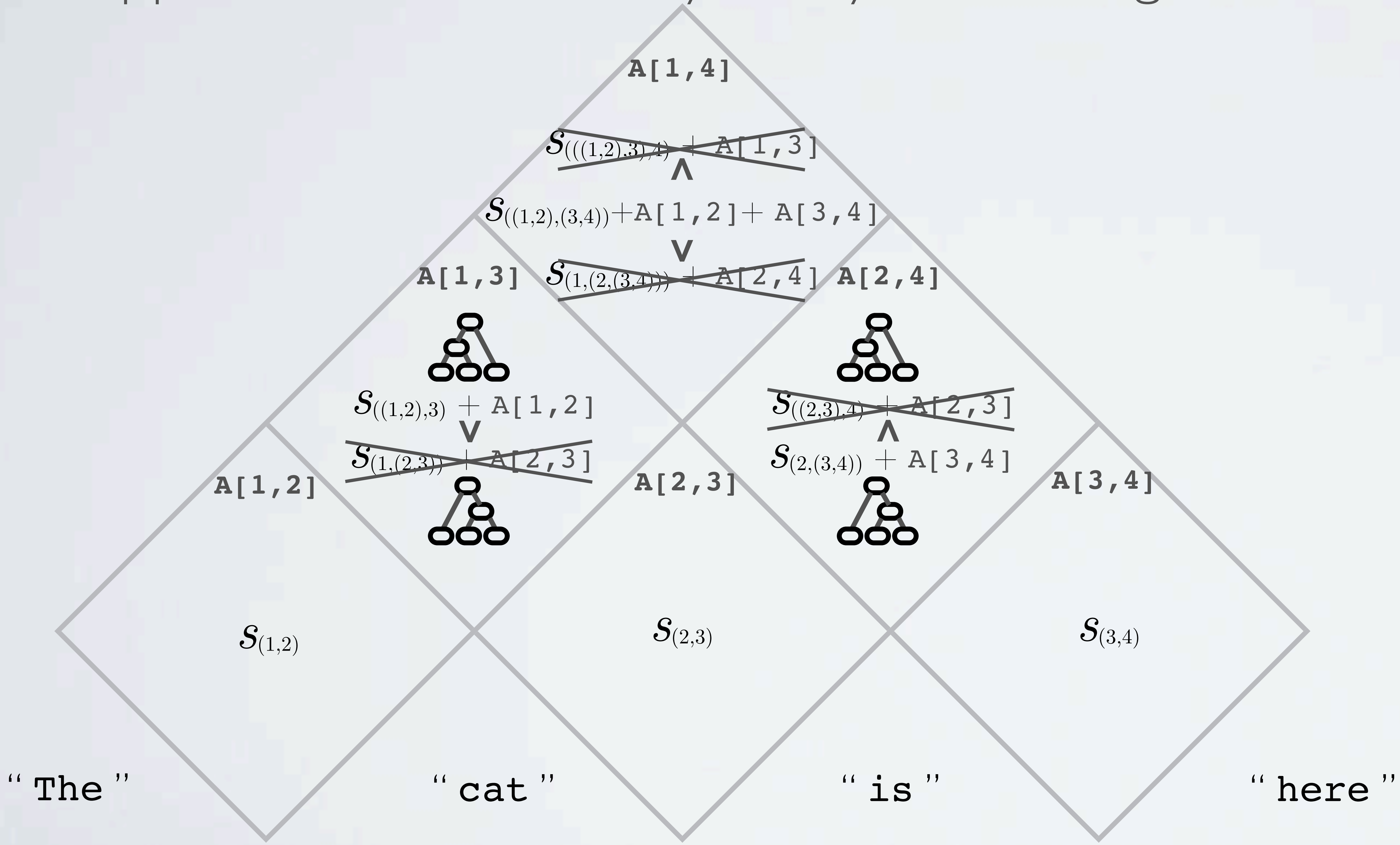
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

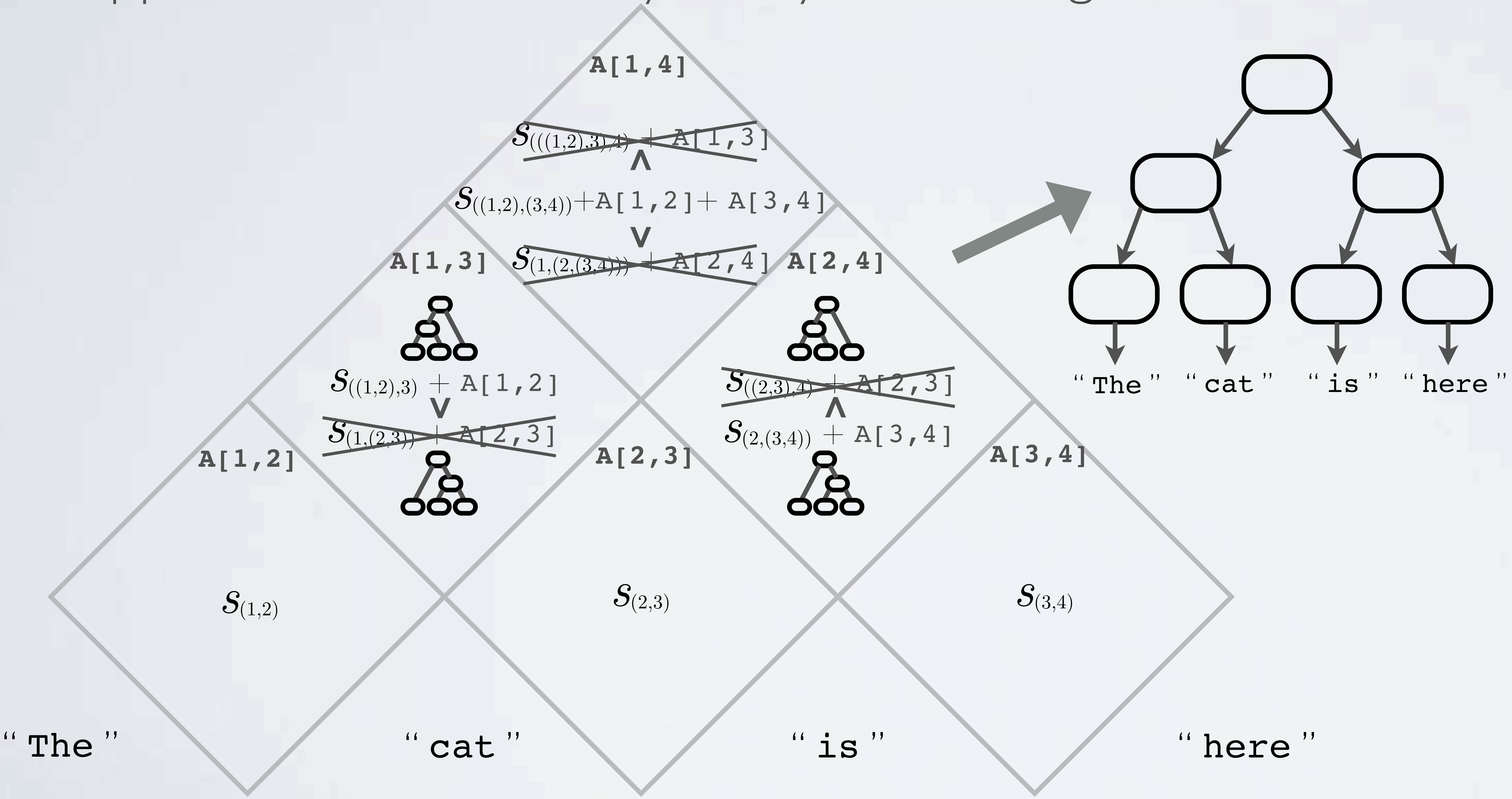
- Approximate best tree by locally maximizing each subtree



RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Approximate best tree by locally maximizing each subtree



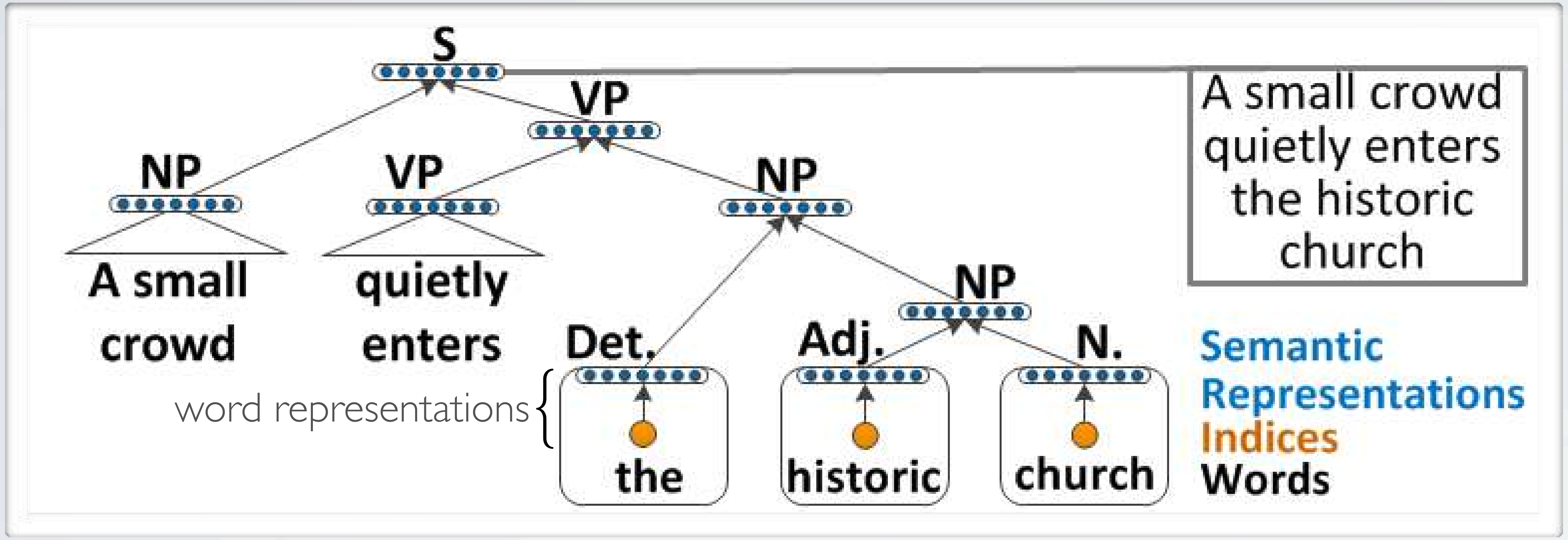
Neural networks

Natural language processing - recursive network training

RECURSIVE NEURAL NETWORK

Topics: recursive neural network (RNN)

- Idea: recursively merge pairs of word/phrase representations



Socher, Lin, Ng and Manning, 2011

- We need 2 things
  - a model that merges pairs of representations
  - a model that determines the tree structure

RECURSIVE NEURAL NETWORK

Topics: training algorithm

- Let y be the true parse tree and $\hat{y}$ be the predicted parse tree
    - we would like the score $s(y)$ of y to be higher than the score $s(\hat{y})$ of $\hat{y}$ (unless $\hat{y}$ is actually y)
- To update the recursive network
    - infer the predicted parse tree $\hat{y}$
    - increase the score $s(y)$ and decrease the score $s(\hat{y})$ by doing an update in the direction of the gradient

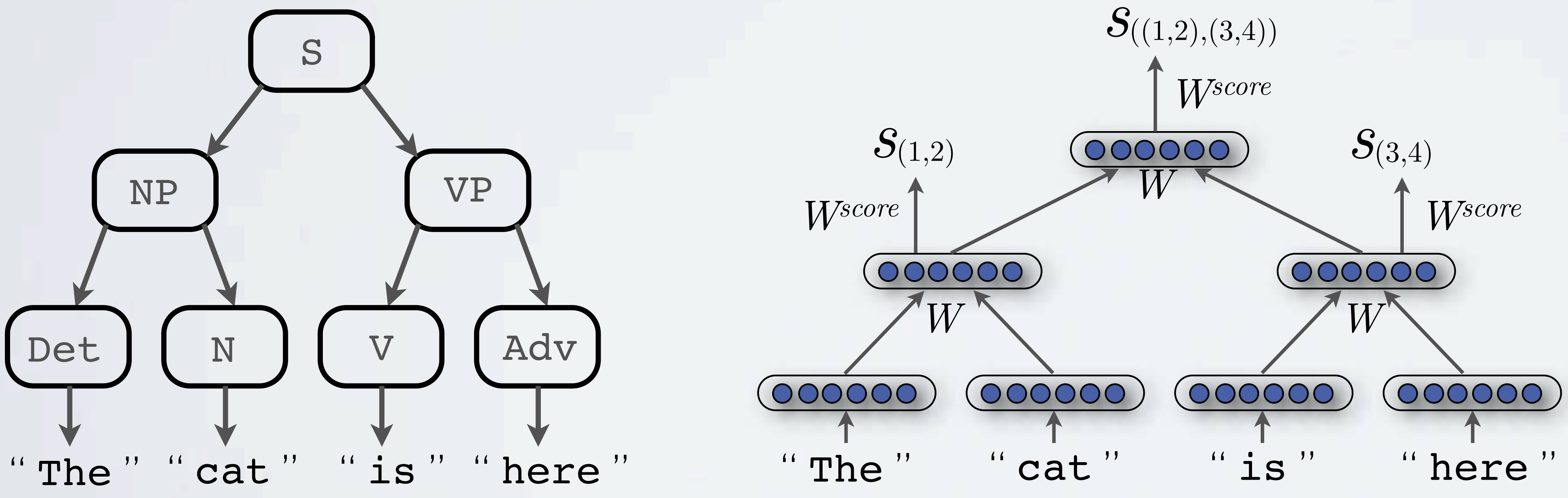
$$\nabla_\theta s(y) - \nabla_\theta s(\hat{y})$$

  - these gradient can be computed by backpropagating through the recursive network structured according to the parse trees y and $\hat{y}$

RECURSIVE NEURAL NETWORK

Topics: training algorithm

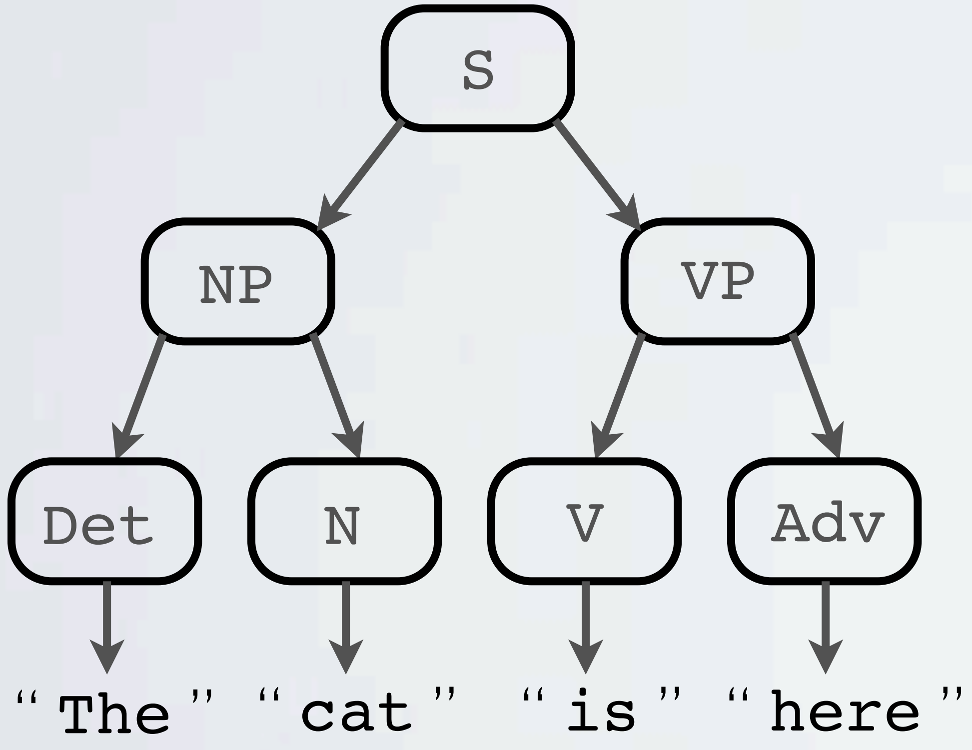
- The nodes of a parse tree are also labeled
  - noun phrase (NP), verb phrase (VP), etc.
  - can add softmax layer that predict the label from each node representation
  - this is an additional gradient to backpropagate, for the true parse tree y

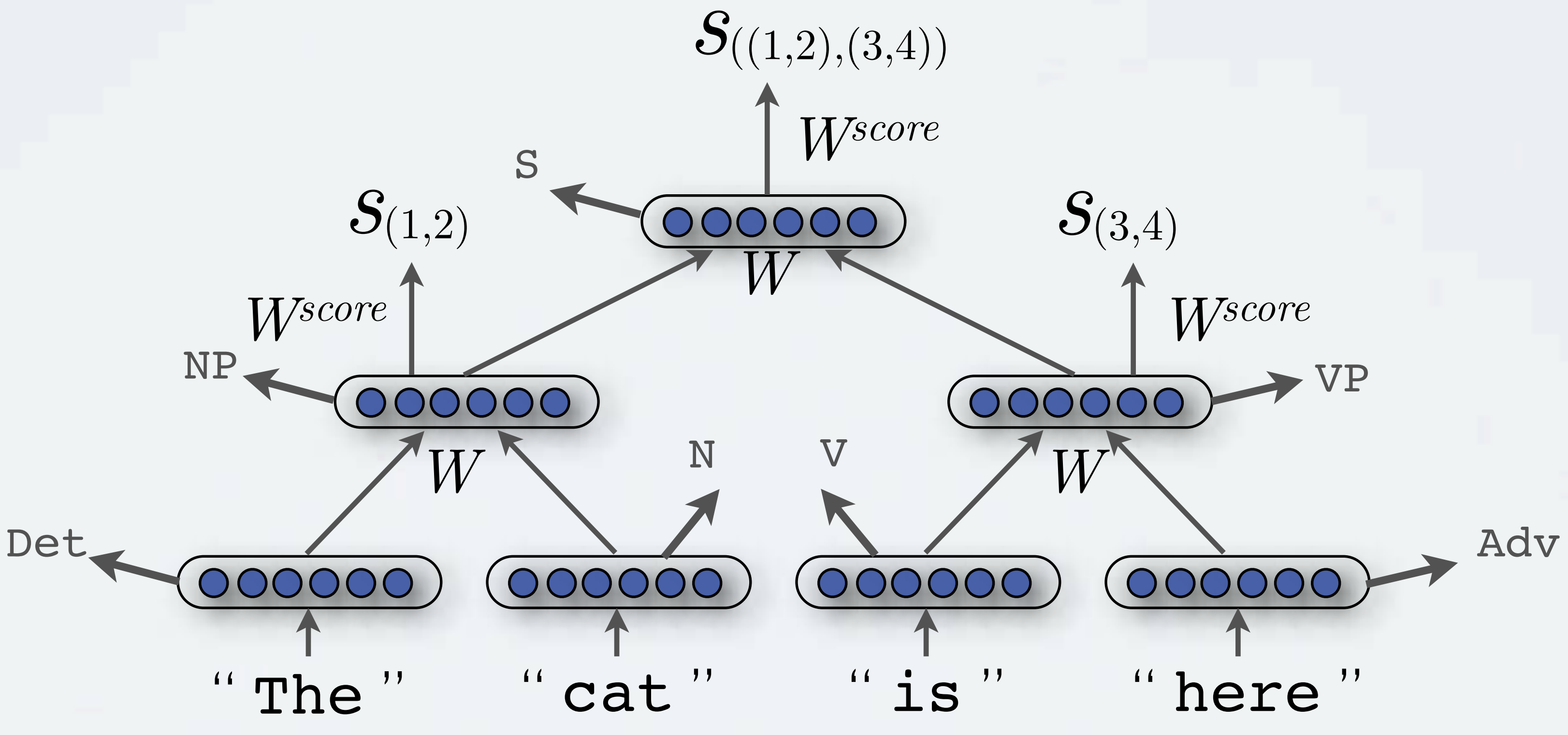


RECURSIVE NEURAL NETWORK

Topics: training algorithm

- The nodes of a parse tree are also labeled
  - noun phrase (NP), verb phrase (VP), etc.
  - can add softmax layer that predict the label from each node representation
  - this is an additional gradient to backpropagate, for the true parse tree y

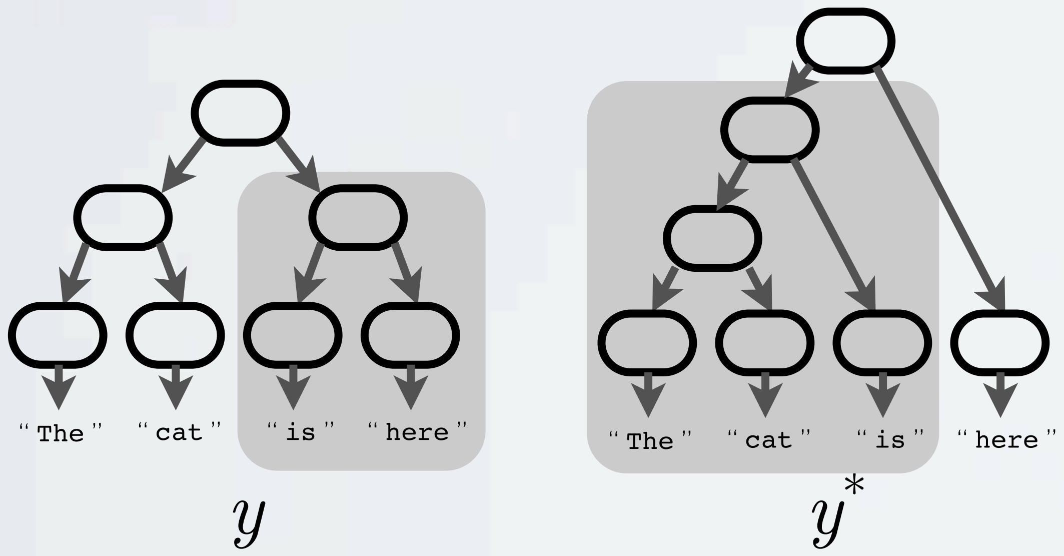




RECURSIVE NEURAL NETWORK

Topics: training algorithm

- Other details
  - word representations are pre-trained using Collobert and Weston's approach and fine-tuned while training the recursive network
  - training is actually based on a margin criteria: $s(y) > s(y^*) + \Delta(y,y^*)$
    - score of the true parse tree y trained to be larger than score of any other tree y^* plus its number of incorrect spans $\Delta(y,y^*)$



number of incorrect span $\Delta(y,y^*)$ $= \; 1$

    - a simple modification to the beam search finding the best tree (see Socher et al. for details)

RECURSIVE NEURAL NETWORK

Topics: experimental comparison

- Parsing F1 performance
  - recursive neural network: 90.29%
  - Berkeley parser: 91.63%

- Nearest neighbor phrases based on RNN representation

Fujisawa gained 50 to UNK		The dollar dropped	
1.	Mead gained 1 to 37 UNK	1.	The dollar retreated
2.	Ogden gained 1 UNK to 32	2.	The dollar gained
3.	Kellogg surged 4 UNK to 7	3.	Bond prices rallied

Socher, Lin, Ng and Manning, 2011