End-To-End Natural Language Processing

Ronan Collobert

ronan@collobert.com

Jason Weston, Léon Bottou, Pavel Kuksa, Koray Kavukcuoglu

NEC Laboratories America

The Goal

We want to have a conversation with our computer

The Goal

 We want to have a conversation with our computer

 Opinion, sentiment analysis

 Business profile

 Semantic search

 Question answering, call center

 Machine Translation

Natural Language Processing + End-To-End Learning

Natural Language Processing

 Part-Of-Speech Tagging (POS): syntactic roles (noun, adverb...)

 Chunking: syntactic constituents (noun phrase, verb phrase...)

 Name Entity Recognition (NER): person/company/location...

 Semantic Role Labeling (SRL): semantic role

[John]$_{ARG0}$ [ate]$_{REL}$ [the apple]$_{ARG1}$ [in the garden]$_{ARGM-LOC}$

How Large-Scale Is It By The Way?

 Part-Of-Speech Tagging (POS): syntactic roles (noun, adverb...)

 Chunking: syntactic constituents (noun phrase, verb phrase...)

 Name Entity Recognition (NER): person/company/location...

 Semantic Role Labeling (SRL): semantic role

[John]$_{ARG0}$ [ate]$_{REL}$ [the apple]$_{ARG1}$ [in the garden]$_{ARGM-LOC}$

Labeled data: Wall Street Journal ($\sim 1M$ of words)
Unlabeled data: Infinite

SVMs with 1M of Labeled Examples

Linear SVMs

- ⋆ Stochastic Gradient Descent see e.g. Bottou
- ⋆ SVMPerf Joachims, 2005
- ⋆ Pegasos Shalev-Shwartz, Singer & Srebro, 2007
- ⋆ LibLinear Lin, Weng & Keerthi, 2008

It can handle it

Non-linear SVMs

- ⋆ LaSVM, 8M of examples. 8 days. Loosli, Canu & Bottou, 2007
  Unfortunately: 150K support vectors! (a non-noisy task...)
- ⋆ Non-convex SVMs (Collobert, Weston & Bottou, 2007) can reduce drastically the number of SVs in a noisy situation
- ⋆ LaSVM+non-convex?

Even if we could do it, non-linear SVMs are slow at testing time

SVMs with ∞ Unlabeled Examples


Most Transductive SVM algorithms can handle only toys


Linear Transductive SVMs

- SVMLin, 5M unlabeled examples, 15 minutes.
  Sindhwani, Keerthi, 2007


Non-Linear Transductive SVMs

  - CCCP-TSVM, 60K unlabeled examples, 42 hours.
    Collobert, Weston & Bottou, 2007.


Any online Transductive SVMs?

Large Scale = Complex Models

In general, if a lot of data is available a complex system is necessary, because the task is complex...

Large Scale = Complex Models


In general, if a lot of data is available a complex system is necessary


Two extreme choices to get a complex system


Large Scale Engineering: design a lot of complex features, use a fast existing linear machine learning algorithm


Large Scale Machine-Learning: use simple features, design a complex model which will implicitly learn the right features

Solution in the middle?

NLP: Large Scale Engineering (1/2)

Choose some good hand designed features

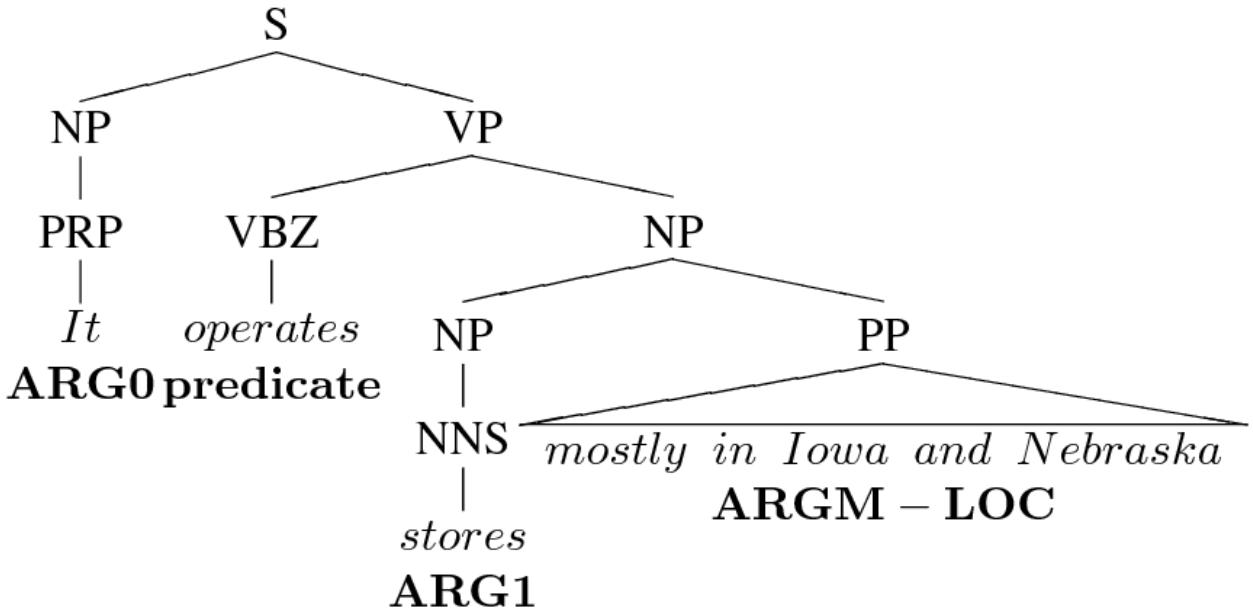
Predicate and POS tag of predicate	Voice: active or passive (hand-built rules)
Phrase type: adverbial phrase, prepositional phrase, . . .	Governing category: Parent node's phrase type(s)
Head word and POS tag of the head word	Position: left or right of verb
Path: traversal from predicate to constituent	Predicted named entity class
Word-sense disambiguation of the verb	Verb clustering
Length of the target constituent (number of words)	NEG feature: whether the verb chunk has a "not"
Partial Path: lowest common ancestor in path	Head word replacement in prepositional phrases
First and last words and POS in constituents	Ordinal position from predicate + constituent type
Constituent tree distance	Temporal cue words (hand-built rules)
Dynamic class context: previous node labels	Constituent relative features: phrase type
Constituent relative features: head word	Constituent relative features: head word POS
Constituent relative features: siblings	Number of pirates existing in the world. . .

Feed them to a shallow classifier like SVM

NLP: Large Scale Engineering (2/2)

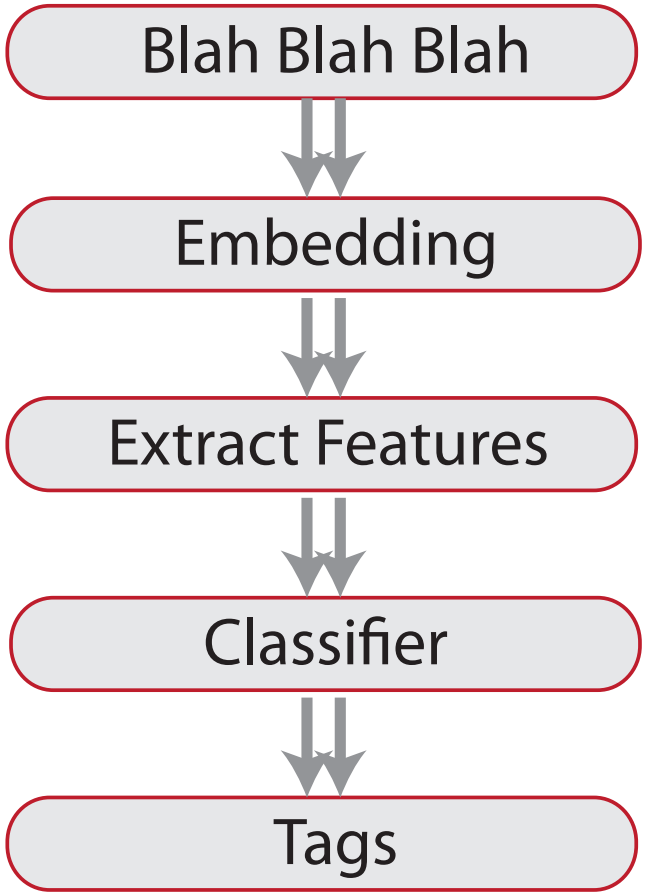
SRL State-of-the-art: the ASSERT system

- Cascade features: e.g. extract POS, construct a parse tree



- Extract hand-made features from the parse tree
- Feed these features to a shallow classifier like SVM

NLP: Large Scale Learning (1/2)



 **Deep** architecture

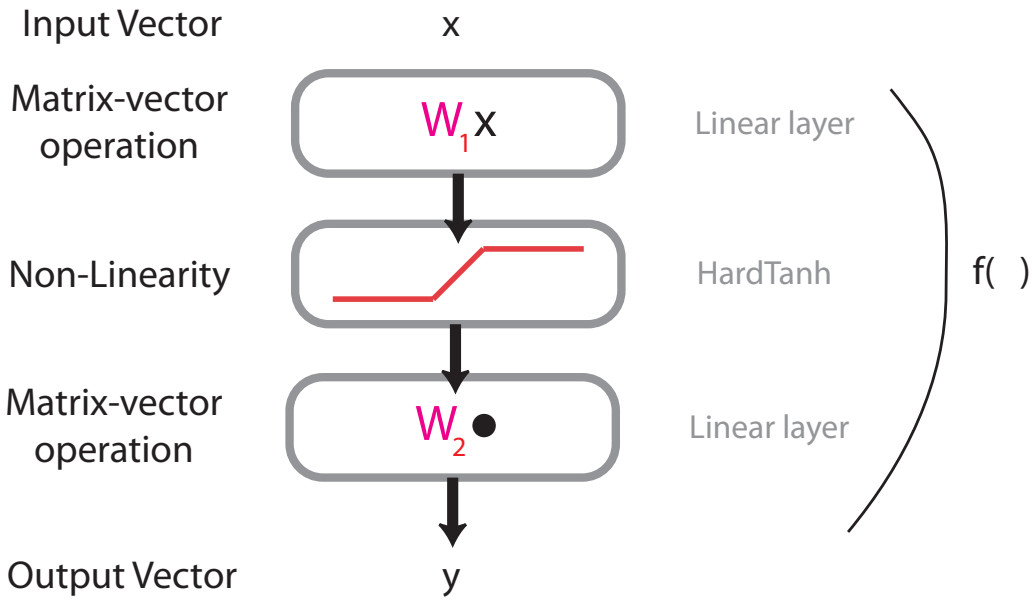
 **Unification** of NLP tasks

NLP: Large Scale Learning (2/2)

Joint training

Neural Networks

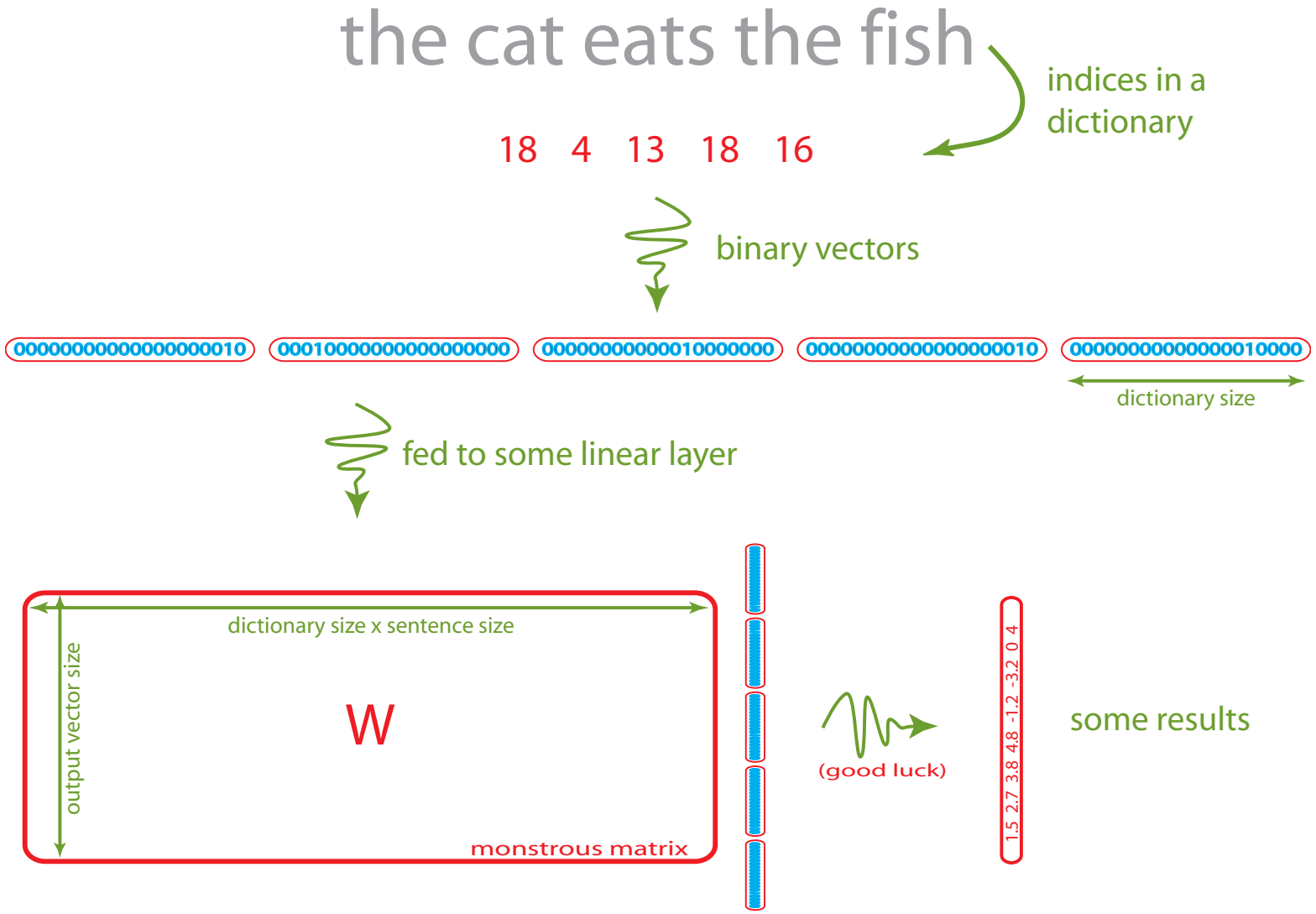
Stack several layers together



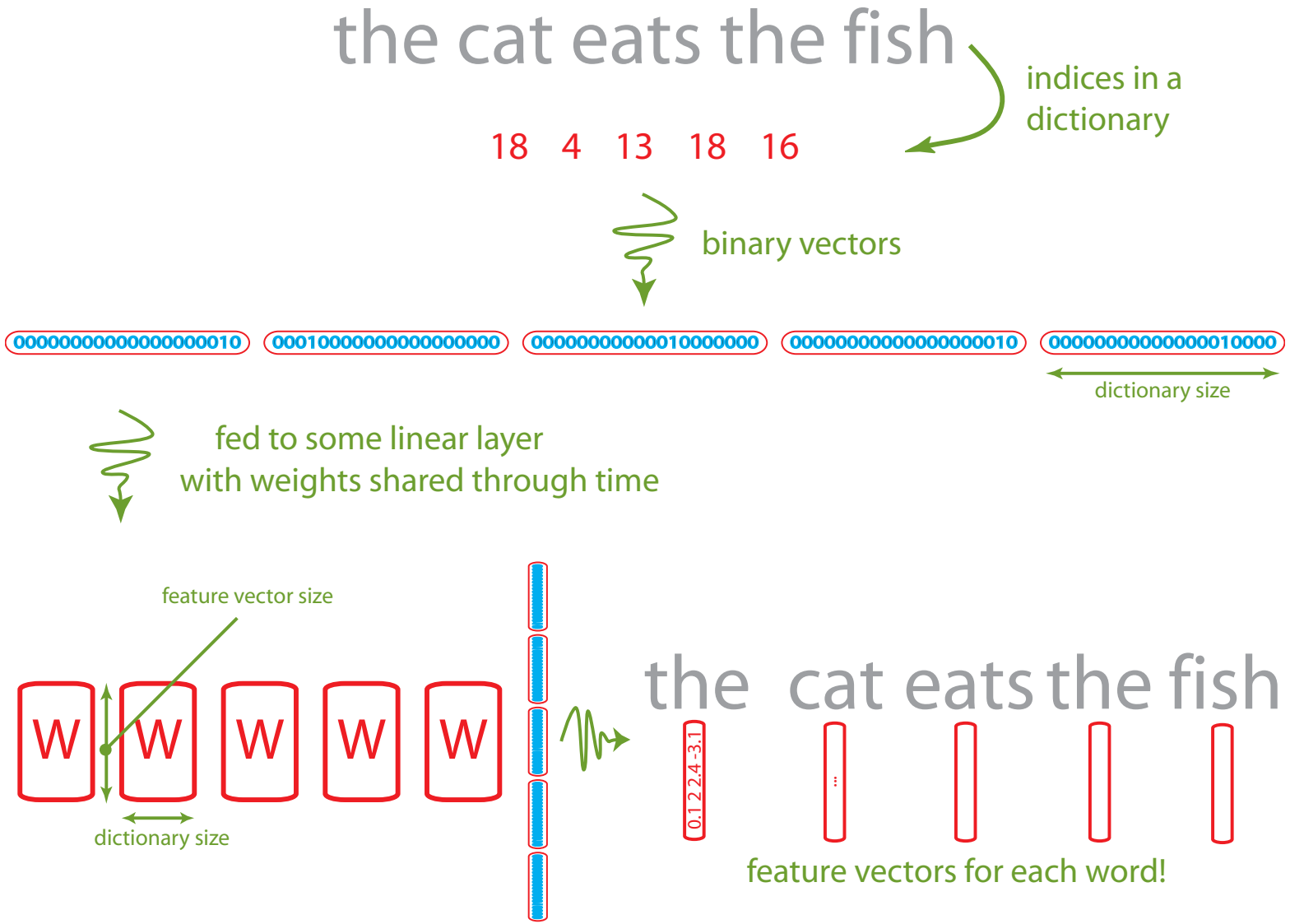
Stochastic gradient descent over a given cost $C()$:

$$W \longleftarrow W - \lambda \frac{\delta C(f(x), y)}{\delta W}$$

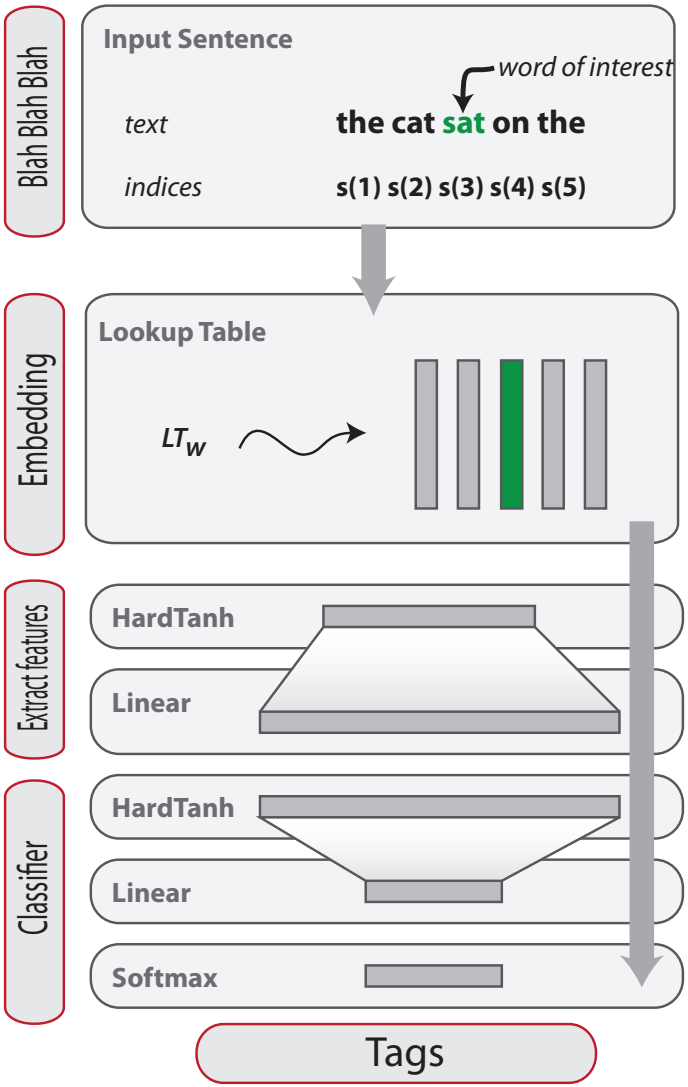
Words into Vectors (1/2)



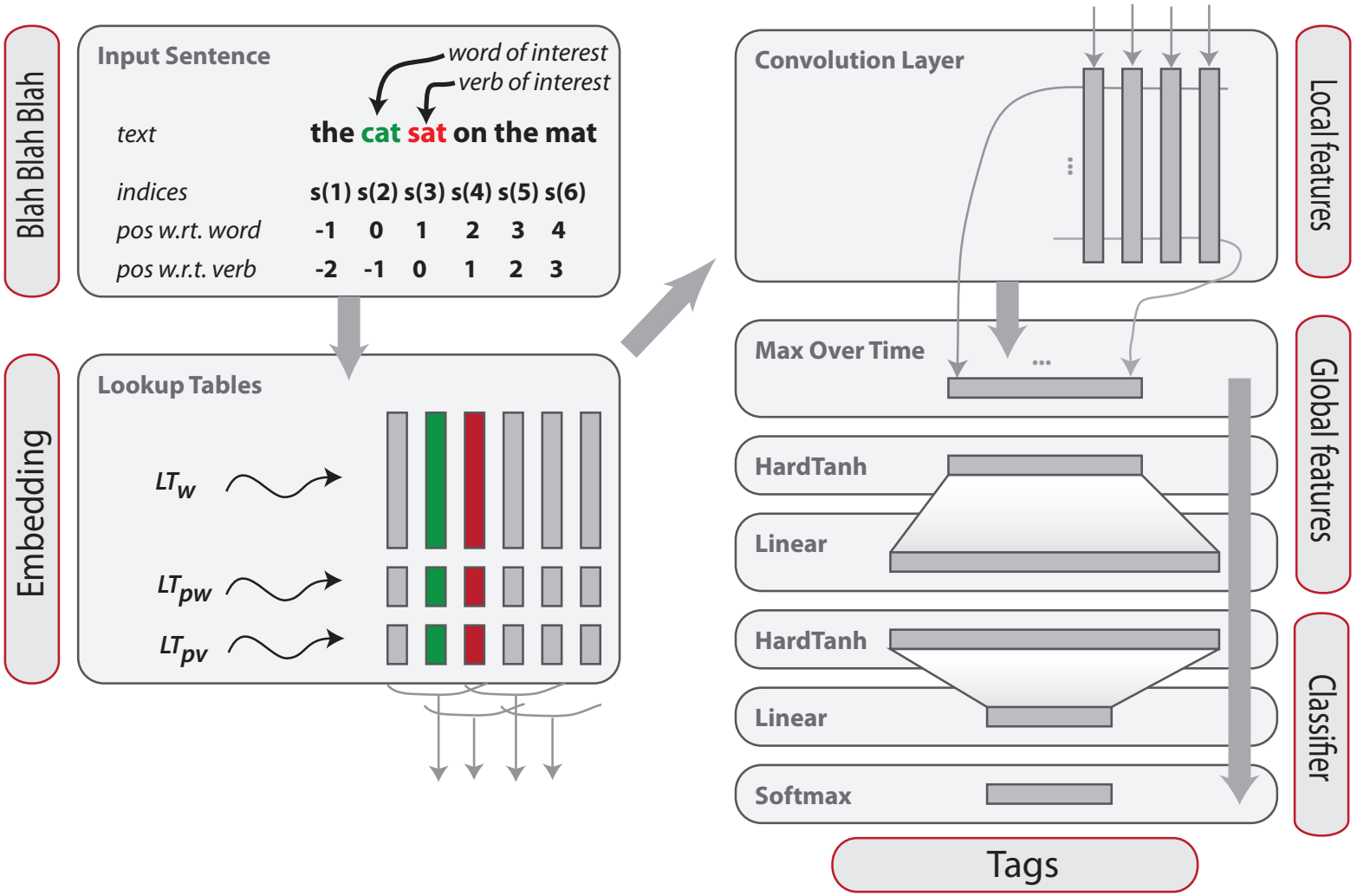
Words into Vectors (2/2)



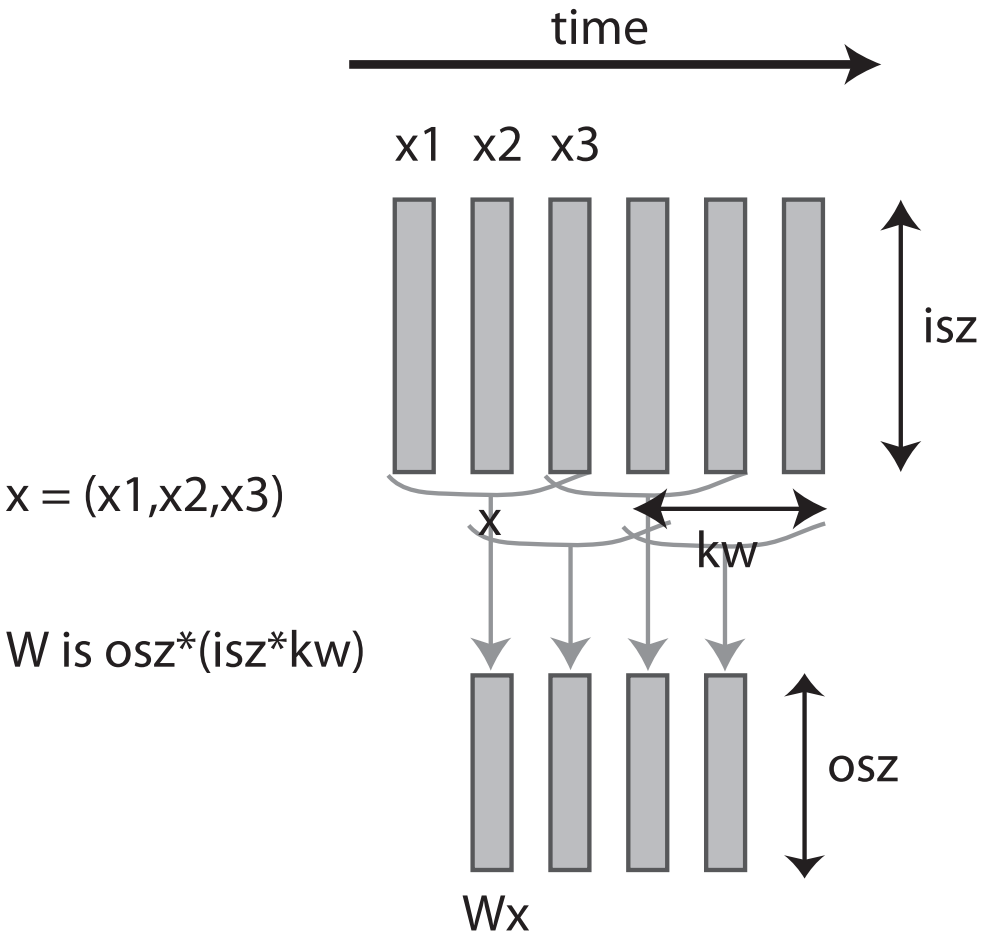
The Deep Learning Way (1/2)



The Deep Learning Way (2/2)



Convolutions

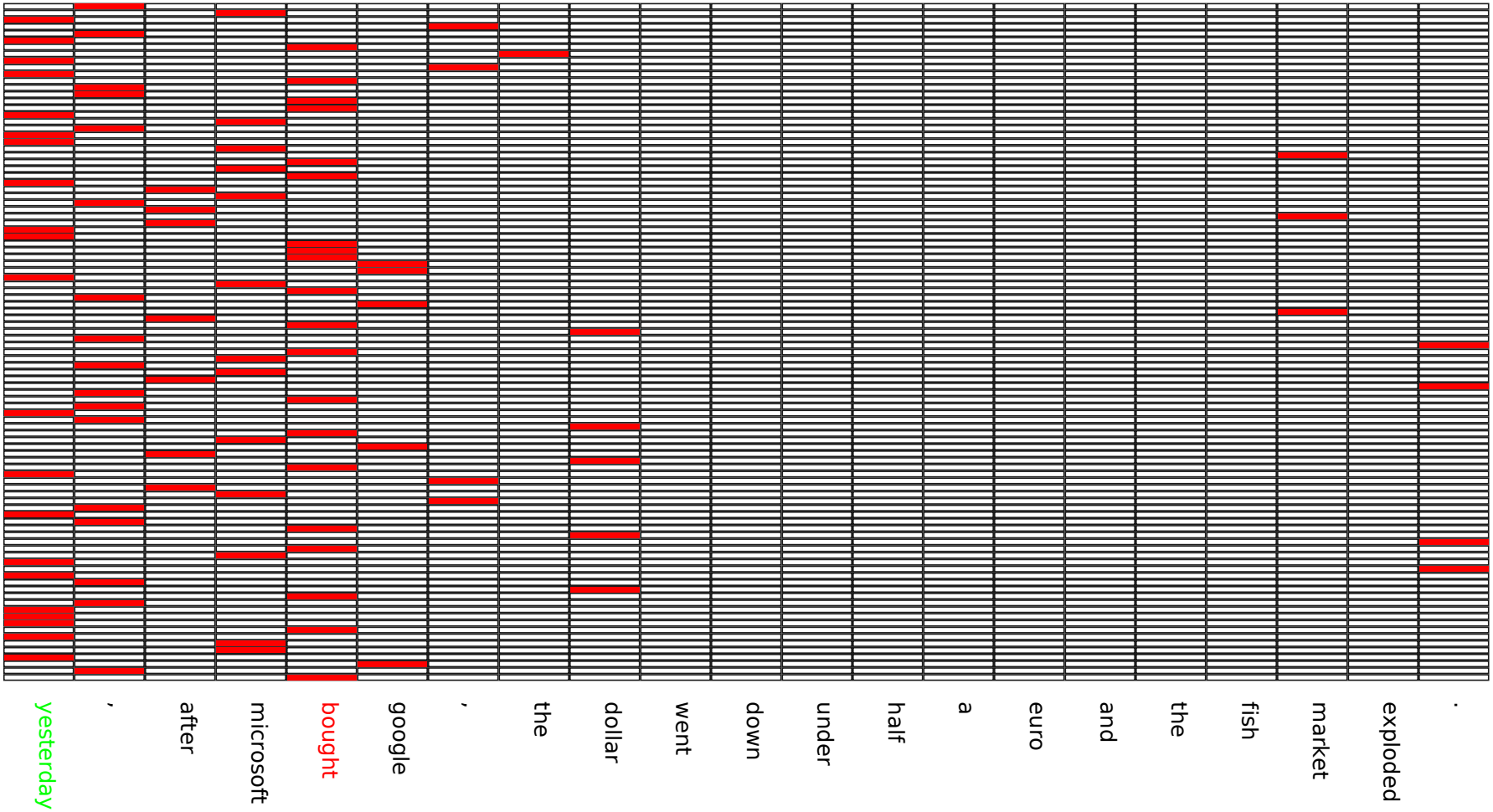


- Extract local features – share weights through time/space
- Used with success in image (Le Cun, 1989) and speech (Bottou & Haffner, 1989)
- Lookup-table is a special case: convolution with kernel size of 1 and input i^{th} word

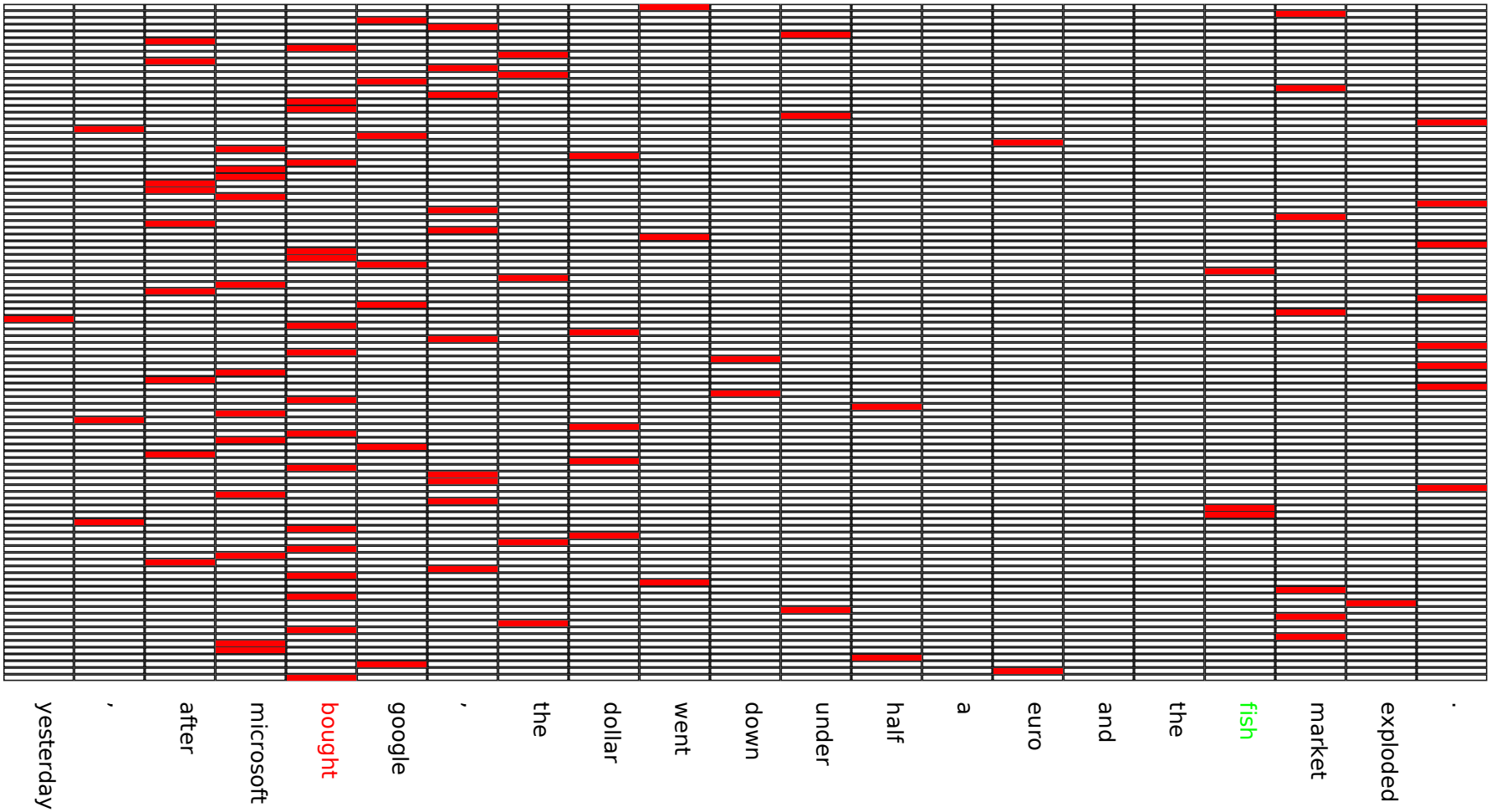
  $(0, 0, \ldots, 1, 0, \ldots, 0)$ 1 at position i

  Bengio et al (2001)

Removing The Time Dimension (1/2)



Removing The Time Dimension (2/2)



Results

WSJ for POS, CHUNK (CoNLL 2000) & SRL (CoNLL 2005)
Reuters (CoNLL 2003) for NER

Approach	POS (% Err)	CHUNK (F1)	NER (F1)	SRL (% Err)
Top Systems	2.76	94.20	88.76	13.36
NN	3.15	88.82	81.61	16.40

Top Systems:
Toutanova et al. (2003) for POS
CoNLL Challenge for NER, Sha et al. (2003) for CHUNK
Punyakanok et al. (2005) for SRL

NN:
Window NN approach for POS, CHUNK & NER
Convolutional NN for SRL
Features: Words, capital letters (+ words suffixes for POS)

1M of Words is not Large Scale Enough!


Dictionary size of WSJ: about 36,000 words. Contains 1M of words.


15% of most frequent words in the dictionary are seen 90% of the time.


Possible improvements:

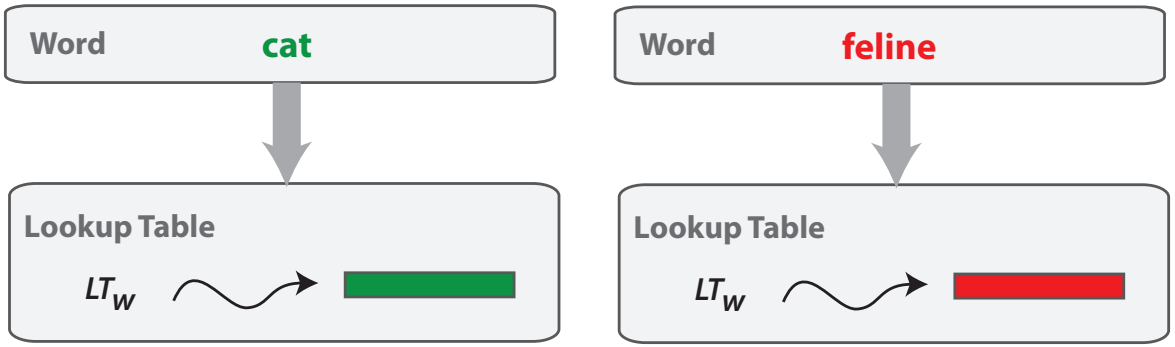

Word clustering (according to POS for e.g.).
See Collobert & Weston, 2007


Thresholding the number of words in the dictionary

- order the words by frequency
- words above the threshold are mapped to a special word "UNKNOWN"

Improving Word Embedding

- Rare words are not trained properly
- Sentences with similar words should be tagged in the same way:
  - The cat sat on the mat
  - The feline sat on the mat



Language Model: Think Massive

Language Model: "*is a sentence actually english or not?*"
Implicitly captures: ⋆ syntax ⋆ semantics

Bengio & Ducharme (2001) Probability of next word given previous words. Overcomplicated – we do not need probabilities here

English sentence windows: Wikipedia **(∼ 631M words)**
Non-english sentence windows: middle word randomly replaced

Ranking margin cost:

$$\sum_{s \in \mathcal{S}} \sum_{w \in \mathcal{D}} \max\left(0,\, 1 - f(s,\, w_s^\star) + f(s,\, w)\right)$$

$\mathcal{S}$: sentence windows $\mathcal{D}$: dictionary
$w_s^\star$: true middle word in s
$f(s,\, w)$: network score for sentence s and middle word w

Language Model: Embedding

france	jesus	xbox	reddish	scratched
454	1973	6909	11724	29869
spain	christ	playstation	yellowish	smashed
italy	god	dreamcast	greenish	ripped
russia	resurrection	psNUMBER	brownish	brushed
poland	prayer	snes	bluish	hurled
england	yahweh	wii	creamy	grabbed
denmark	josephus	nes	whitish	tossed
germany	moses	nintendo	blackish	squeezed
portugal	sin	gamecube	silvery	blasted
sweden	heaven	psp	greyish	tangled
austria	salvation	amiga	paler	slashed

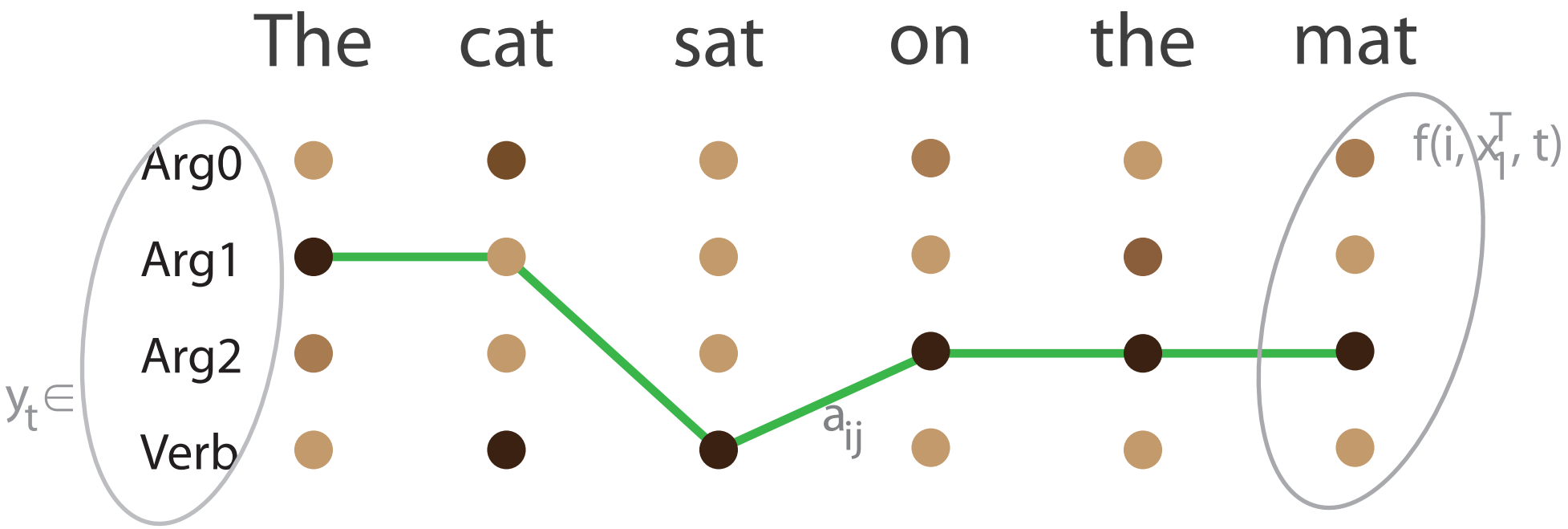
Dictionary size: 30,000 words. Even rare words are well embedded.

Language Model: Results

Approach	POS (% Err)	CHUNK (F1)	NER (F1)	SRL (% Err)
Top Systems	2.76	94.20	88.76	13.36
NN	3.15	88.82	81.61	16.40
NN+LM	2.80	91.87	86.57	15.13

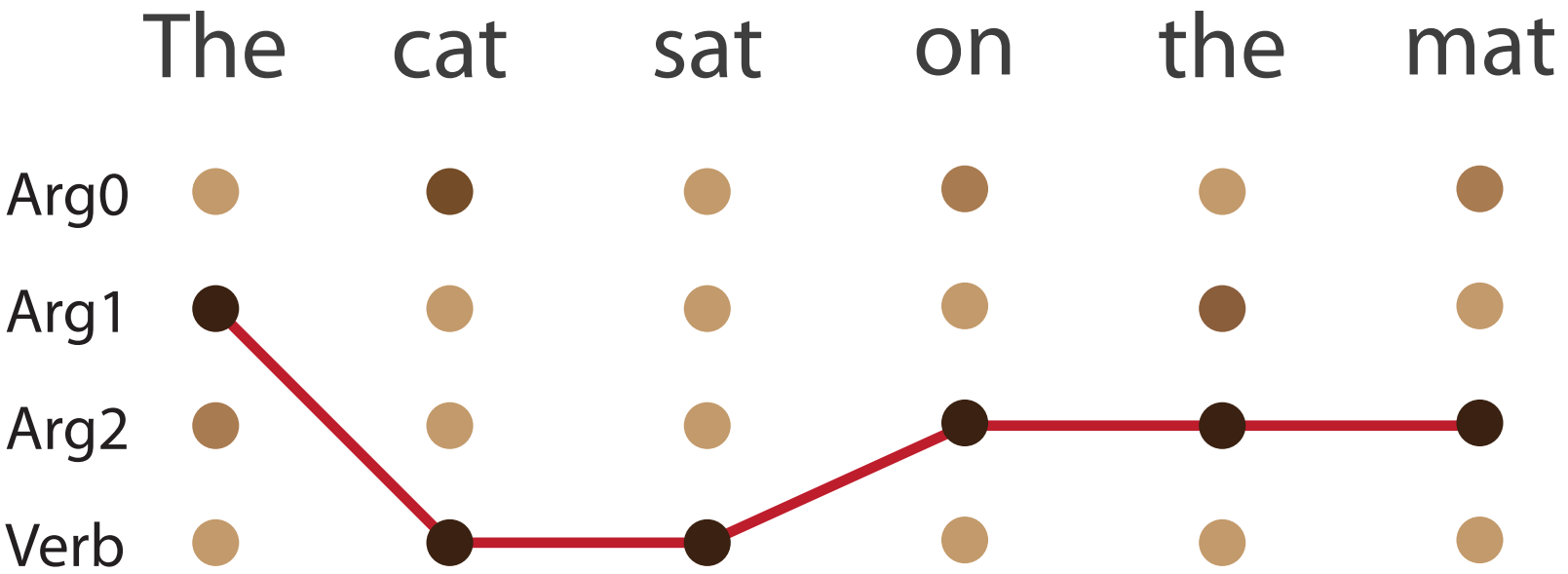
Structured Output Learning (1/2)

- ⋆ We consider a observation sequence x_1^T and a label sequence y_1^T
- ⋆ Network outputs score $f_\theta(i, x_1^T, t)$ at time t for each label $y_t = i$ given x_1^T
- ⋆ Consider a transition score to jump from label i to j equal to $a_{i,j}$



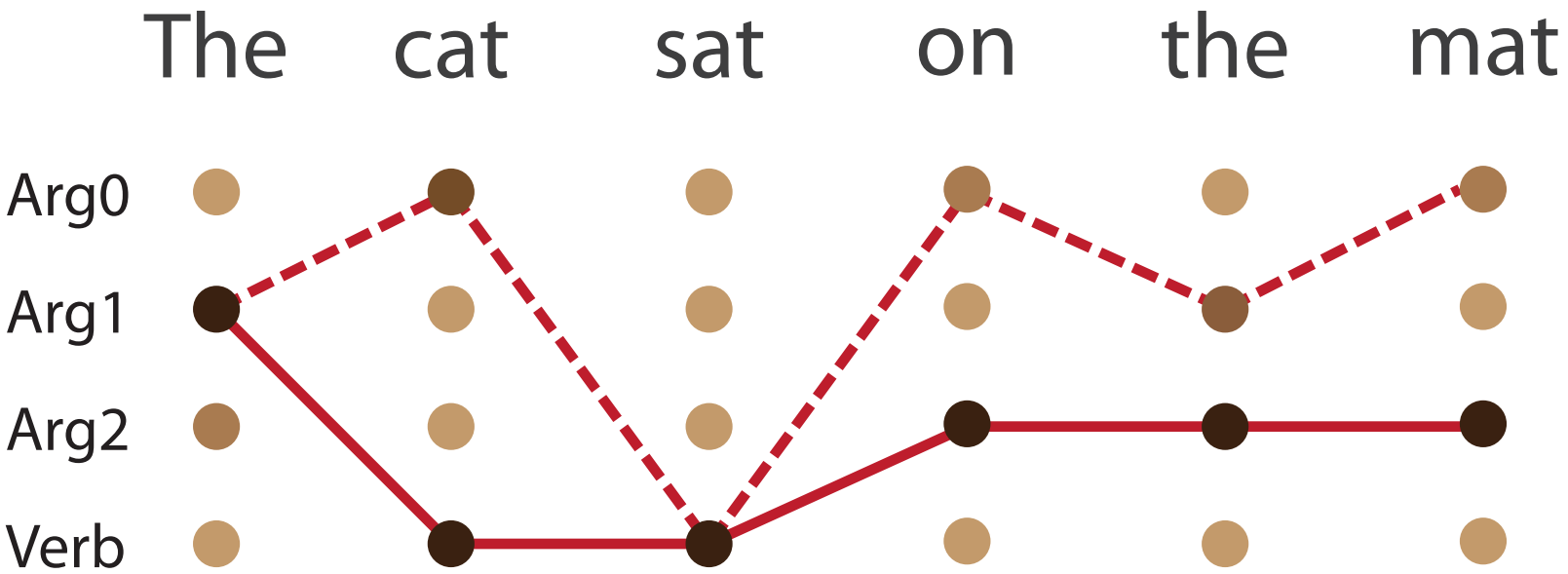
Structured Output Learning (1/2)

- ⋆ We consider a observation sequence x_1^T and a label sequence y_1^T
- ⋆ Network outputs score $f_\theta(i, x_1^T, t)$ at time t for each label $y_t = i$ given x_1^T
- ⋆ Consider a transition score to jump from label i to j equal to $a_{i,j}$



Structured Output Learning (1/2)

- ⋆ We consider a observation sequence x_1^T and a label sequence y_1^T
- ⋆ Network outputs score $f_\theta(i, x_1^T, t)$ at time t for each label $y_t = i$ given x_1^T
- ⋆ Consider a transition score to jump from label i to j equal to $a_{i,j}$



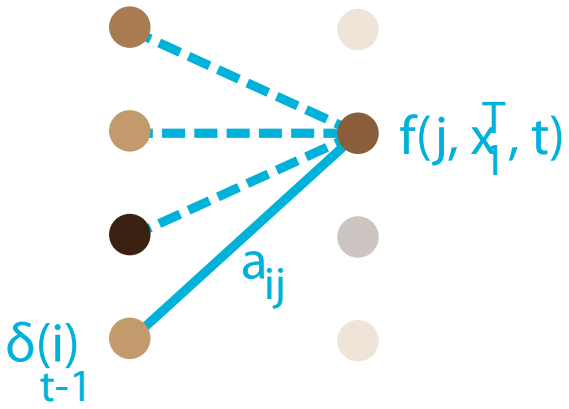
Structured Output Learning (2/2)

Generalization of CRFs: Graph Transformer Networks, LeCun (1998)

- Maximize the difference of forward scores

$$F(x_1^T, \theta)_{|y_1^T \text{ is correct}} - F(x_1^T, \theta)$$

- Forward score $F(x_1^T, \theta)$ recursively defined:



$$\delta_t(j) = \text{logAdd}_i \left[\delta_{t-1}(i) + a_{i,j} + f_\theta(j, x_1^T, t)\right]$$

Termination:

$$F(x_1^T, \theta) = \text{logAdd}_i \, \delta_T(i)$$

Where $\text{logAdd}(a+b) = \log\left(e^a + e^b\right)$

- Inference: Viterbi algorithm (replace logAdd by max)

Structured Output Learning: Results

Approach	POS (% Err)	CHUNK (F1)	NER (F1)	SRL (% Err)
Top Systems	2.76	94.20	88.76	13.36
NN	3.15	88.82	81.61	16.40
NN+LM	2.80	91.87	86.57	15.13
NN+SOL	–	90.08	83.43	–
NN+LM+SOL	–	93.78	88.52/87.25†	13.82*

$\dagger$ no gazetteer

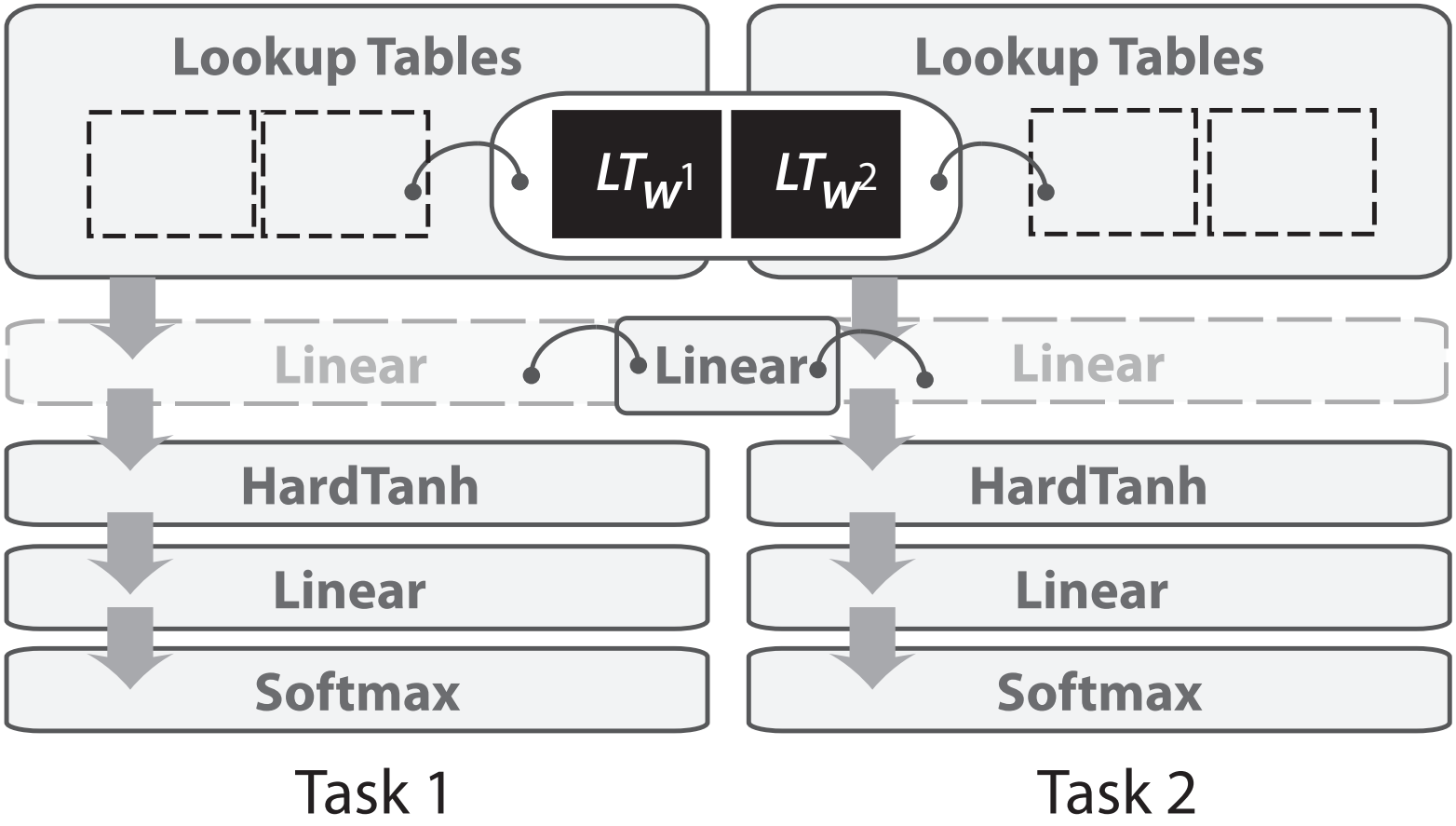
$\star$ 200x faster than state-of-the-art for SRL

Structured Output Learning: Results

Approach	POS (% Err)	CHUNK (F1)	NER (F1)	SRL (% Err)
Top Systems	2.76	94.20	88.76	13.36
NN	3.15	88.82	81.61	16.40
NN+LM	2.80	91.87	86.57	15.13
NN+SOL	–	90.08	83.43	–
NN+LM+SOL	–	93.78	88.52/87.25†	13.82
NN+LM+POS+SOL	–	94.18	88.22	–

† no gazetteer

Multi-Task Learning



Good overview in Caruana (1997)

Multi-Task Learning: Results

Approach	POS (% Err)	CHUNK (F1)	NER (F1)	SRL (% Err)
Top Systems	2.76	94.20	88.76	13.36
NN	3.15	88.82	81.61	16.40
NN+LM	2.80	91.87	86.57	15.13
NN+SOL	–	90.08	83.43	–
NN+LM+SOL	–	93.78	88.52	13.82
NN+LM+POS+SOL	–	94.18	88.22	–
NN+LM+MTL	2.80	92.44	86.22	–
NN+LM+SOL+MTL	2.80	94.13	88.10	–

Summary


We developed a deep neural network architecture for NLP


Advantages

- ⋆ General to any NLP tagging task
- ⋆ State-of-the-art performance
- ⋆ No hand designed features
- ⋆ Joint training
- ⋆ Can exploit massive unlabeled data
- ⋆ Extremely fast: 0.02s for all tags of a sentence


Inconvenients

- ⋆ Neural networks are a powerful tool: easy to mess up


Early Impacts

- ⋆ Easy extension to other tasks/languages: Japanese & German
- ⋆ Fast: developed a semantic search system