Natural Language Processing (almost) from Scratch

Ronan Collobert

Jason Weston

Léon Bottou

Michael Karlen

Koray Kavukcuoglu

Pavel Kuksa

Speaker: Jason Chen

Part-of-speech tagging (POS)

Chunking (CHUNK)

Named entity recognition (NER)

Semantic Role Labeling (SRL)

Motivation & Benchmark Test

Part-of-speech tagging

Chunking

Named entity recognition

Semantic Role Labeling

- Labeling each word with an unique tag that indicates its syntactic role.
- E.g. Noun Phrases (NP), Verb Phrases (VP), Determiner (Det), etc..

& Motivation Benchmark Test

Part-of-speech tagging

Chunking

Named entity recognition

Semantic Role Labeling

Motivation & Benchmark Test

- Labeling segments of a sentence with syntactic constituents such as noun or verb phrases (NP or VP).
- Each word is assigned only one unique tag
- E.g. B-NP for begin-chunk tag, I-NP for inside-chunk tag.

Part-of-speech tagging

Chunking

Named entity recognition

Semantic Role Labeling

- Labeling atomic elements in the sentence into categories.
- E.g. PERSON, LOCATION, DATE, NUMERIC

Motivation & Benchmark Test

Part-of-speech tagging

Chunking

Named entity recognition

Semantic Role Labeling

Motivation & Benchmark Test

- Giving a semantic role to a syntactic constituent of a sentence.
- Feature categories include:
  - The POS and syntactic labels of words;
  - The Node's position in relation to the verb;
  - The syntactic path to the verb in the parse tree;
  - The verb sub-categorization;
  - The voice of the sentence: active or passive.
- E.g. John ate the apple

John	ate	the apple
ARG0	REL	ARG1

Neural Network Structure

Traditional Approach

1. Extract a rich set of hand designed feature from sentence.
2. Feed the feature set into classification algorithm, such as Support Vector Machine (SVM) with a linear kernel.

Neural Network Approach

1. Takes the feature vectors of complete sentence/segment of text (window).
2. Passes through the lookup table layer, transform words into feature vectors.
3. Produces local features around each word of the sentence using linear/convolutional layers.
4. Combine local features into a global feature vector.
5. Feed into standard affine layers.

Transform Words into Feature Vectors

Cat

Feature 1	w_1^1
Feature 2	w_1^2
$\vdots$	$\vdots$
Feature k	w_1^k

- Maps each word indices into feature vectors by a look up table operation.
- Consider of efficiency, words are feeding as indices.
- Formally noted as

$$\mathrm{LT}_{\mathrm{W}}(w) = \langle W \rangle_{w}^{1}, \ w \in \mathcal{D}$$

- The above equation can be extend as

$$\mathrm{LT}_{\mathrm{W}}([w]_1^T) = \langle W \rangle_{w}^{1}$$

Transform Words into Feature Vectors - Extend

Cat

Vector 1

$$\begin{matrix} \text{Feature } 1 & w_1^1 \\ \text{Feature } 2 & w_1^2 \\ \vdots & \vdots \\ \text{Feature } k & w_1^k \end{matrix}$$

Vector 2

$$\begin{matrix} \text{Feature } 1 & w_1^1 \\ \text{Feature } 2 & w_1^2 \\ \vdots & \vdots \\ \text{Feature } k & w_1^k \end{matrix}$$

- Extend to any discrete features, provide features other than words if one suspects that these features.
- E.g. pre-processing keeps case information.
- Formally noted as

$$\mathrm{LT}_{W^1,\dots,\mathrm{W}^\mathrm{K}}(w) = \begin{pmatrix} \langle W \rangle_{w_K}^1 \\ \dots \\ \langle W \rangle_{w_K}^1 \end{pmatrix}$$

- The above equation can be extend as

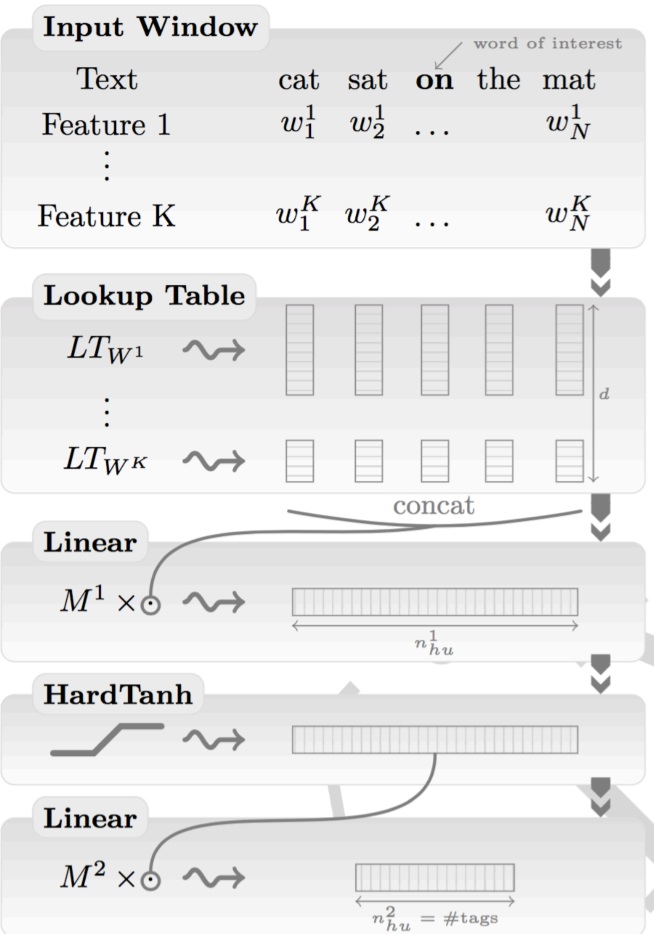
$$\mathrm{LT}_{W^1,\dots,\mathrm{W}^\mathrm{K}}([w]_1^T) = \begin{pmatrix} \langle W \rangle_{[w_1]_1}^1 & \cdots & \langle W \rangle_{[w_1]_T}^1 \\ \vdots & \ddots & \vdots \\ \langle W \rangle_{[w_K]_1}^1 & \cdots & \langle W \rangle_{[w_K]_T}^1 \end{pmatrix}$$

1

Extract High Level Features from Word Feature Vector

②

Extract High Level Features from Word Feature Vector – Window Approach

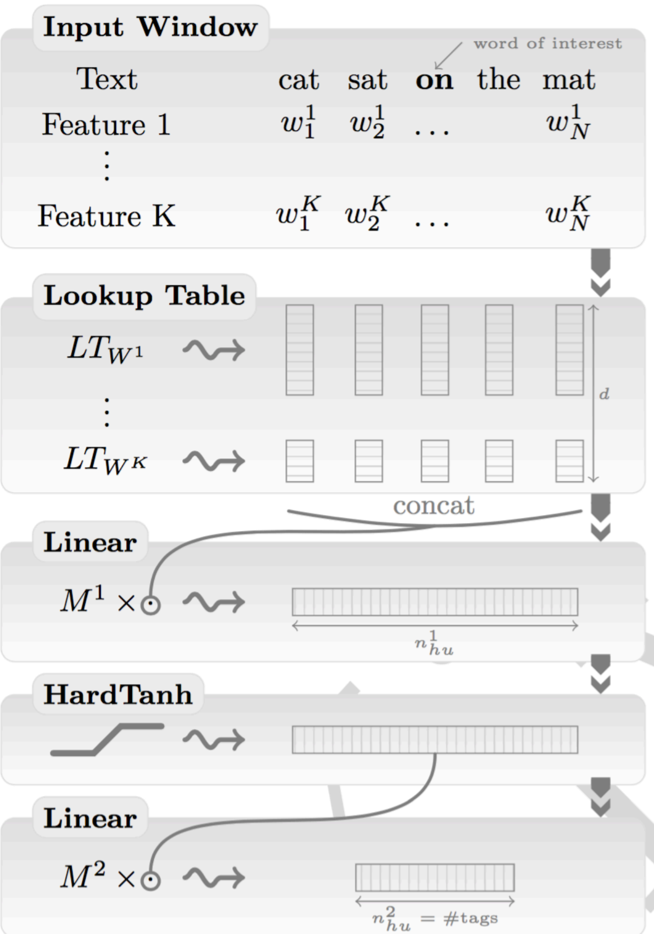


- Variable length equals to width of the window k_{sz}.
- Given a word to tag, a fixed size window of words around this word.
- Each window passed through the lookup table layer, producing a word features matrix with size $d_{wrd} \times k_{sz}$

$$f_\theta^1 = \langle \mathrm{LT}_W([w]_1^T) \rangle_t^{d_{win}} = \begin{pmatrix} \langle W \rangle_{[w]_{t-d_{win}/2}}^1 \\ \vdots \\ \langle W \rangle_{[w]_t}^1 \\ \vdots \\ \langle W \rangle_{[w]_{t+d_{win}/2}}^1 \end{pmatrix}$$

2

Extract High Level Features from Word Feature Vector – Window Approach



- f_θ^l can feed to one or several standard linear neural network layers.

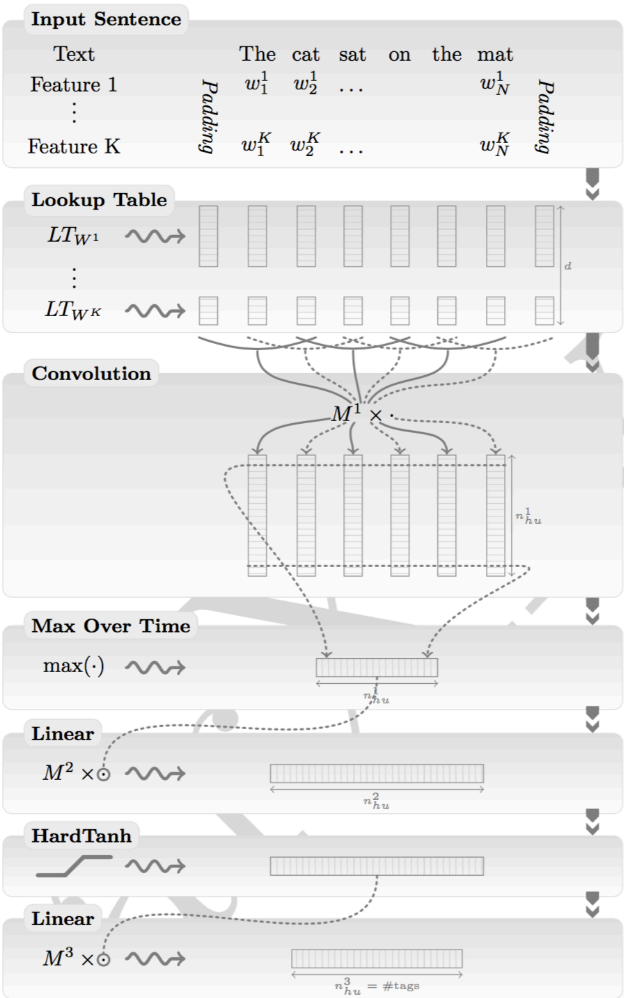
$$f_\theta^l = W^l f_\theta^{l-1} + b^l$$

- Use HardTanh layer as the activation function.

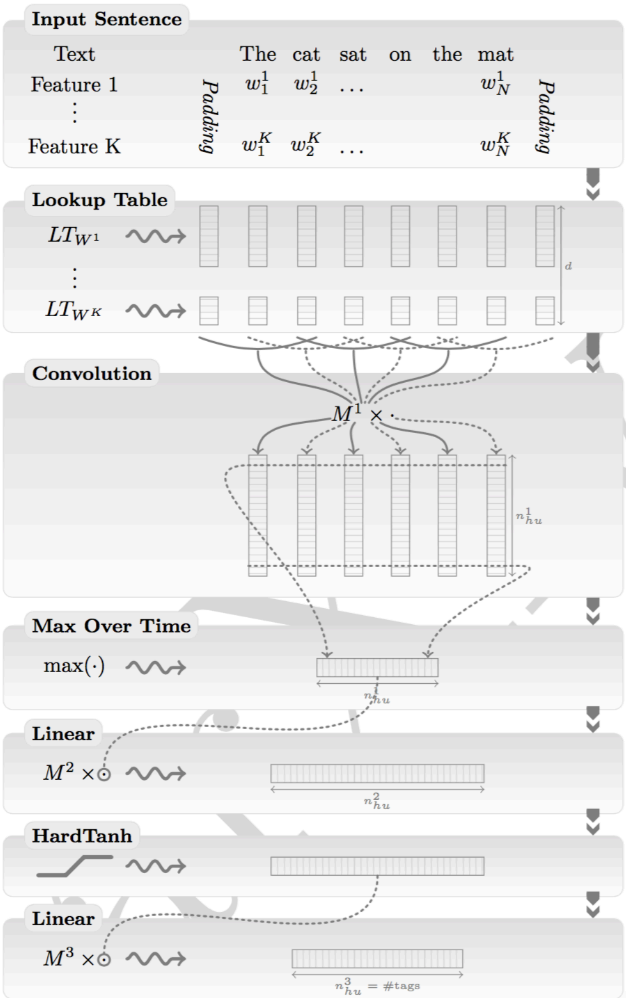
$$\mathrm{HardTanh}(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 \le x \le 1 \\ 1 & \text{if } x > 1 \end{cases}$$

- Padding special "PADDING" word $d_{win}/2$ times at the beginning and the end.

Extract High Level Features from Word Feature Vector – Sentence Approach



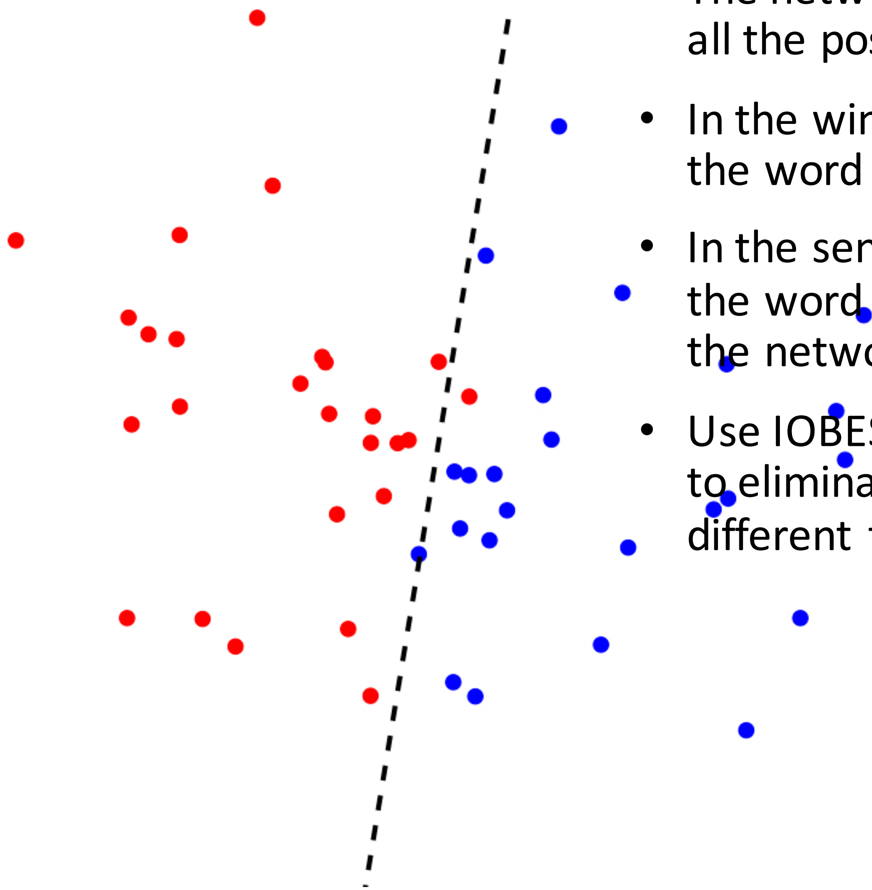
- Window approach fails with SRL, where the tag of a word depends on a verb chosen beforehand in the sentence and the verb falls outside the window.
- In this case, tagging a word requires the consideration of the whole sentence.
- Implementing convolutional layer for sentence approach. A convolutional layer can be seen as a generalization of a window approach.



Extract High Level Features from Word Feature Vector – Sentence Approach

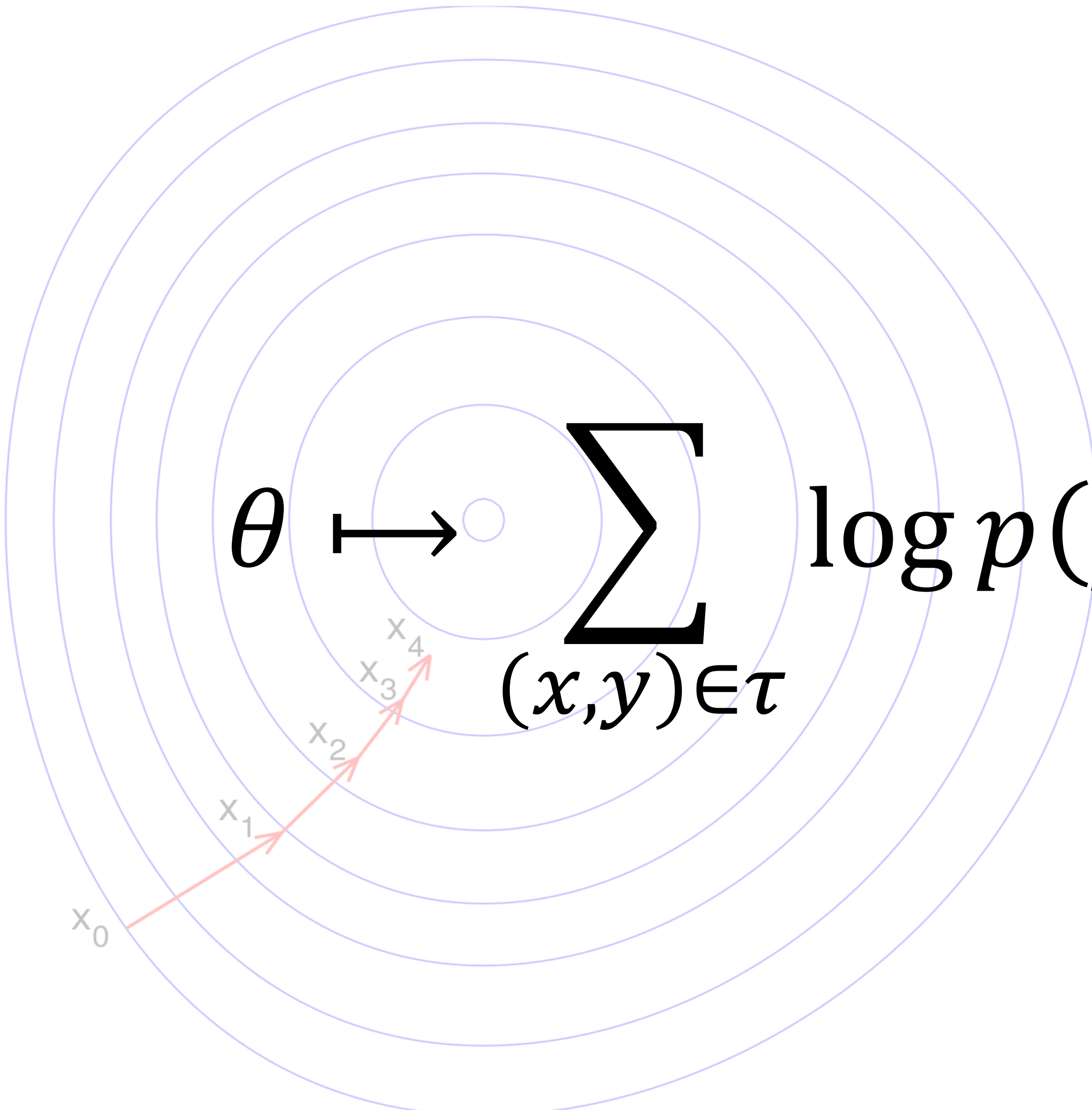
- Formally, using previous notation, the convolutional layer can be noted as
$$\langle f_\theta^l \rangle_t^1 = W^l \langle f_\theta^{l-1} \rangle_t^{d_{win}} + b^l$$
- W^l is shared across all windows t in the sequence.
- Convolutional layers are often stacked to extract higher level features, so, it must be followed a non-linearity layer.
- We use Max Layer here.
$$[f_\theta^l] = \max_t [f_\theta^{l-1}]_{i,t} \quad 1 \le i \le n_{hu}^{l-1}$$

Everything is for Tagging



- The network output layers compute scores for all the possible tags for the task of interest.
- In the window approach, theses tags apply to the word located in the center of the window.
- In the sentence approach, these tags apply to the word designated by additional markers in the network input.
- Use IOBES tagging scheme for all tasks, in order to eliminate additional source of variations that different task using different tagging schemes.

$$\theta \longmapsto \sum_{(x,y)\in\tau} \log p(y|x,\theta)$$



- The log-likelihood can be expressed as

$$\log p(y|x,\theta) = [f_\theta]_y - \log \sum_i e^{[f_\theta]_j}$$

- The score can be interpreted as a conditional tag probability by apply a softmax operation

$$p(i|x,\theta) = \frac{e^{[f_\theta]_i}}{\sum_j e^{[f_\theta]_j}}$$

Training

Word-Level

Sentence-Level

Stochastic Gradient

Benchmark Results

Training

- Finding the best path of tags during training.
- Introducing a transition score $[A]_{i,j}$ for jumping from i to j tags in successive words.
- The new training parameter is
$$\tilde{\theta} = \theta \cup \{[A]_{i,j} \; \forall i,j\}$$
- The score of a sentence along a path of tags is given as
$$s\left([x]_1^T, [i]_1^T, \tilde{\theta}\right) = \sum_{t=1}^{T} \left([A]_{[i]_{t-1},[i]_t} + [f_\theta]_{[i]_t,t}\right)$$
- The probability of true path
$$\log p\left([y]_1^T \middle| [x]_1^T, \tilde{\theta}\right)$$
$$= s\left([x]_1^T, [y]_1^T, \tilde{\theta}\right) - \underset{\forall [j]_1^T}{\operatorname{logadd}}\, s\left([x]_1^T, [j]_1^T, \tilde{\theta}\right)$$
- After mathematical simplification, we find minimizes the sentence score can find best tag path
$$\underset{[j]_1^T}{\operatorname{argmax}}\, s\left([x]_1^T, [j]_1^T, \tilde{\theta}\right)$$

Word-Level

Sentence-Level

Stochastic Gradient

Benchmark Results

$$\theta \leftarrow \theta + \lambda \frac{\partial \log p(y|x, \theta)}{\partial \theta}$$

Training

Word-Level

Sentence-Level

Stochastic Gradient

Benchmark Results

Training

- **Tasks:**
  1. POS
  2. CHUNK
  3. NER
  4. SRL
- **Methods:**
  1. Benchmark Systems
  2. Neural Network + Word-Level Log-Likelihood
  3. Neural Network + Sentence-Level Log-Likelihood
- **Tricks:**
  1. All networks were fed with two raw text features: low case words and capital letter feature.
  2. Number was replaced as NUMBER, words outside the dictionary is replaced as RARE.

Approach	**POS** (PWA)	**Chunking** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99

Task	Window/Conv. size	Word dim.	Caps dim.	Hidden units	Learning rate
POS	$d_{win} = 5$	$d^0 = 50$	$d^1 = 5$	$n^1_{hu} = 300$	$\lambda = 0.01$
CHUNK	”	”	”	”	”
NER	”	”	”	”	”
SRL	”	”	”	$n^1_{hu} = 300$ $n^2_{hu} = 500$	”

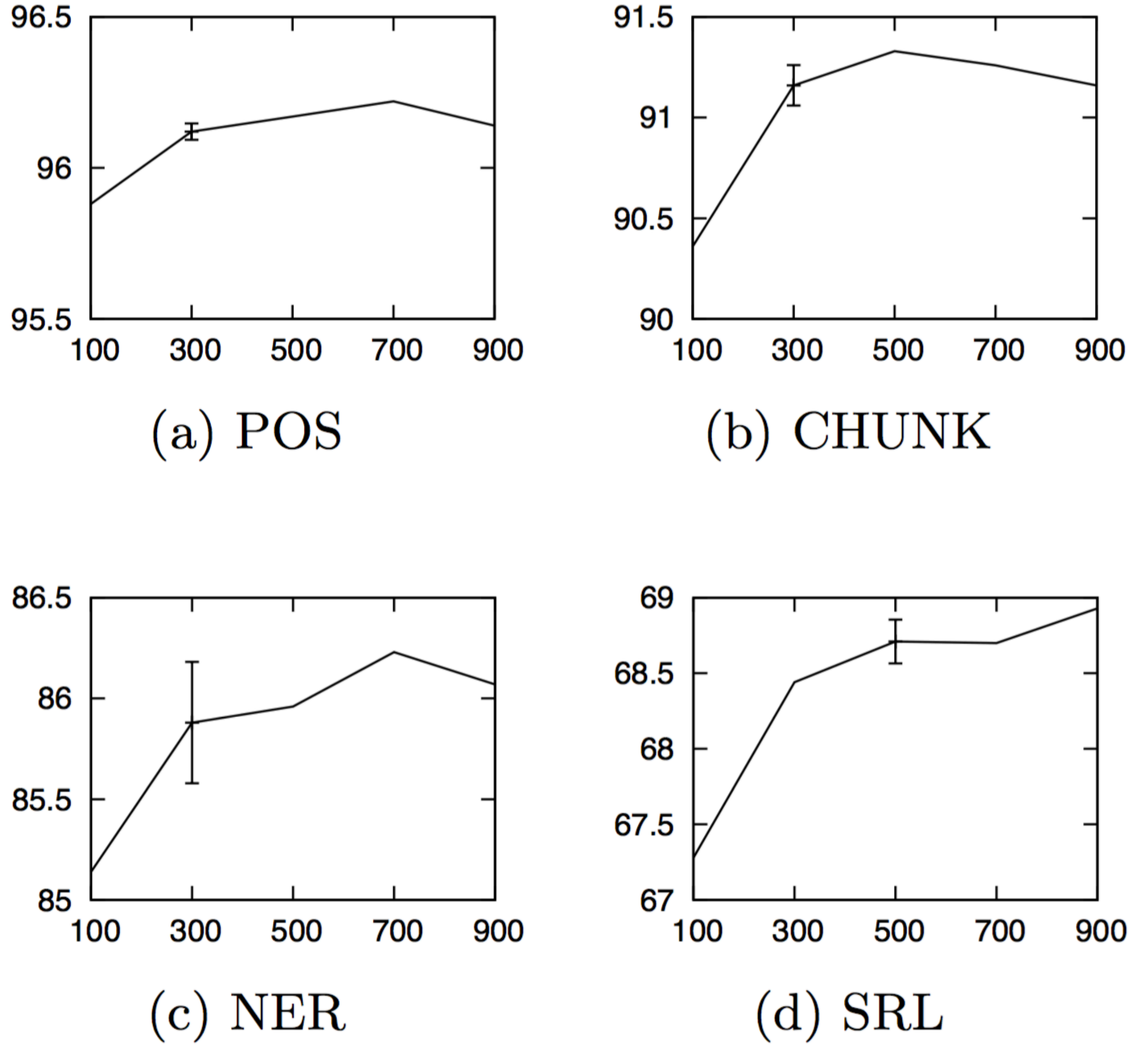
Training

Word-Level

Sentence-Level

Stochastic Gradient

Benchmark Results



Training

Not yet

Improvement

Using unlabeled data

Improvements

Pairwise Criterion

$$\theta \longmapsto \sum_{x \in \mathcal{X}} \sum_{w \in \mathcal{D}} \max\{0, 1 - f_\theta(x) + f_\theta(x^{(w)})\}$$

Improvements

Training Language Models
(breeding)

1. Initially, choose k parameters choices from the set of all possible parameters
2. Select the best ones using the validation set error rate.
3. In next iteration, we choose another set of k parameters from the possible grid of values that permute slightly the most successful candidates from previous round.
4. Repeat 2 and 3.

Benefits: Many of parameter choice can share weights.

Improvements

Language Model LM1

Language model LM1 has a window size $d_{win} = 11$ and a hidden layer with $n_{hu}^{1} = 100$ units. The embedding layers were dimensioned as 50. Model LM1 was trained on our first English corpus (Wikipedia) using successive dictionaries composed of the 5000, 10,000, 30,000, 50,000 and finally 100, 000 most common WSJ words.

Language Model LM2

Based on the word embeddings obtained by LM1, trained on the Wikipedia+Reuters corpus for addition.

Improvements

Old

Neither syntactic nor semantic relationship

FRANCE 454	JESUS 1973	XBOX 6909	REDDISH 11724	SCRATCHED 29869	MEGABITS 87025
PERSUADE	THICKETS	DECADENT	WIDESCREEN	ODD	PPA
FAW	SAVARY	DIVO	ANTICA	ANCHIETA	UDDIN
BLACKSTOCK	SYMPATHETIC	VERUS	SHABBY	EMIGRATION	BIOLOGICALLY
GIORGI	JFK	OXIDE	AWE	MARKING	KAYAK
SHAHEED	KHWARAZM	URBINA	THUD	HEUER	MCLARENS
RUMELIA	STATIONERY	EPOS	OCCUPANT	SAMBHAJI	GLADWIN
PLANUM	ILIAS	EGLINTON	REVISED	WORSHIPPERS	CENTRALLY
GOA'ULD	GSNUMBER	EDGING	LEAVENED	RITSUKO	INDONESIA
COLLATION	OPERATOR	FRG	PANDIONIDAE	LIFELESS	MONEO
BACHA	W.J.	NAMSOS	SHIRT	MAHAN	NILGIRIS

New

The syntactic and semantic are clearly related

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

Improvements

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99
NN+WLL+LM1	97.05	91.91	85.68	58.18
NN+SLL+LM1	97.10	93.65	87.58	73.84
NN+WLL+LM2	97.14	92.04	86.96	58.34
NN+SLL+LM2	97.20	93.63	88.67	74.15

Not yet

Improvement

Using Multi-Task Learning

Improvements

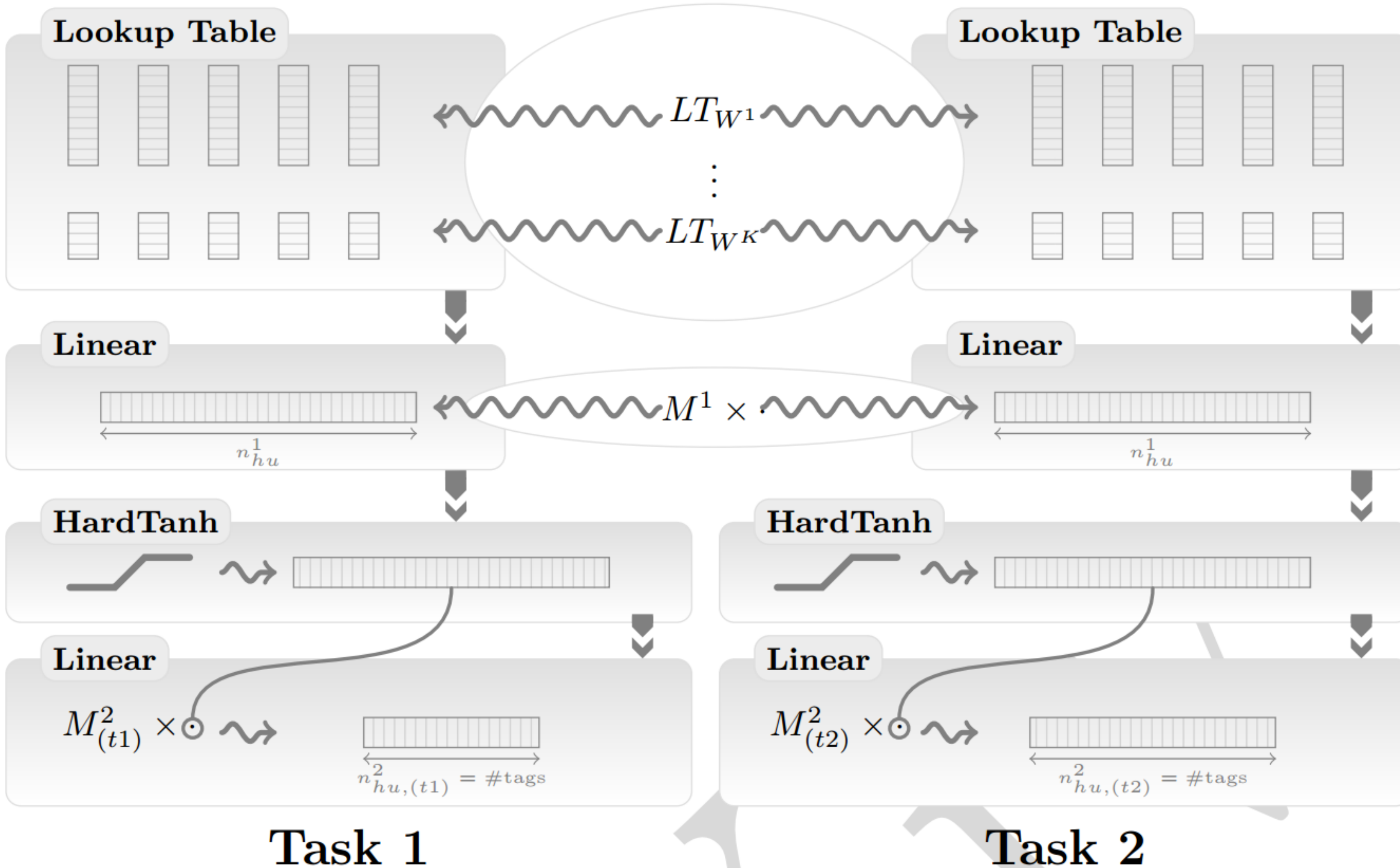
Joint Decoding

1. Considering additional probabilistic dependency paths between the models.
2. Therefor, it defines an implicit supermodel that describes all the tasks in the same probabilistic framework.
3. However, separately training cannot make dependency paths directly involve unobserved variables.

Joint Training

1. Good find relation for the case that training sets for the individual tasks contain the same patterns with different labels.
2. Sufficient to train a model that that computes multiple outputs form each pattern.

Improvements



Improvements

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	–
	Window Approach			
NN+SLL+LM2	97.20	93.63	88.67	–
NN+SLL+LM2+MTL	97.22	94.10	88.62	–
	Sentence Approach			
NN+SLL+LM2	97.12	93.37	88.78	74.15
NN+SLL+LM2+MTL	97.22	93.75	88.27	74.29

Not yet

Improvement

Using Tricks

Improvements

1. **Suffix Features:** Strong predictors of the syntactic function in western languages.
2. **Gazetteers:** Large (8,000) category dictionary of name entity.
3. **Cascading:** Tags obtained for one task might useful for taking decisions in others.
4. **Ensembles:** Use multiple learning algorithms to obtain better performance.
5. **Parsing:** Provide parse tree information as additional input features to the system.
6. **Word Representations:** Induced word embedding on large amount of unlabeled text data.

Improvements

Final Results

Task		**Benchmark**	**SENNA**
Part of Speech (POS)	(Accuracy)	97.24 %	97.29 %
Chunking (CHUNK)	(F1)	94.29 %	94.32 %
Named Entity Recognition (NER)	(F1)	89.31 %	89.59 %
Parse Tree level 0 (PT0)	(F1)	91.94 %	92.25 %
Semantic Role Labeling (SRL)	(F1)	77.92 %	75.49 %

Improvements

Resources Usage

POS System	RAM (MB)	Time (s)
Toutanova et al. (2003)	800	64
Shen et al. (2007)	2200	833
SENNA	32	4

SRL System	RAM (MB)	Time (s)
Koomen et al. (2005)	3400	6253
SENNA	124	51

Conclusion

Q & A

☺ Thanks for your attention ☺