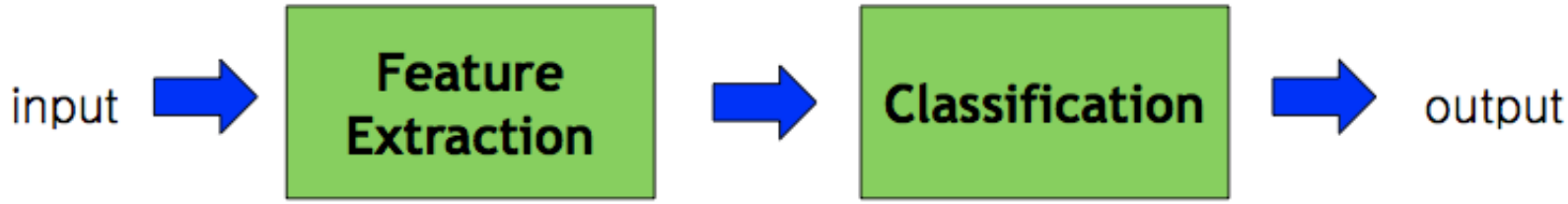
자연어처리를 위한 딥러닝 기술

강원대학교 IT대학
이창기

차례

Why Deep Neural Networks?: **Integrated Learning**

- **기존 기계학습 방법론**
  - **Handcrafting features → time-consuming**



- **Deep Neural Network: Feature Extractor + Classifier**

<겨울학교14 Deep Learning 자료 참고>

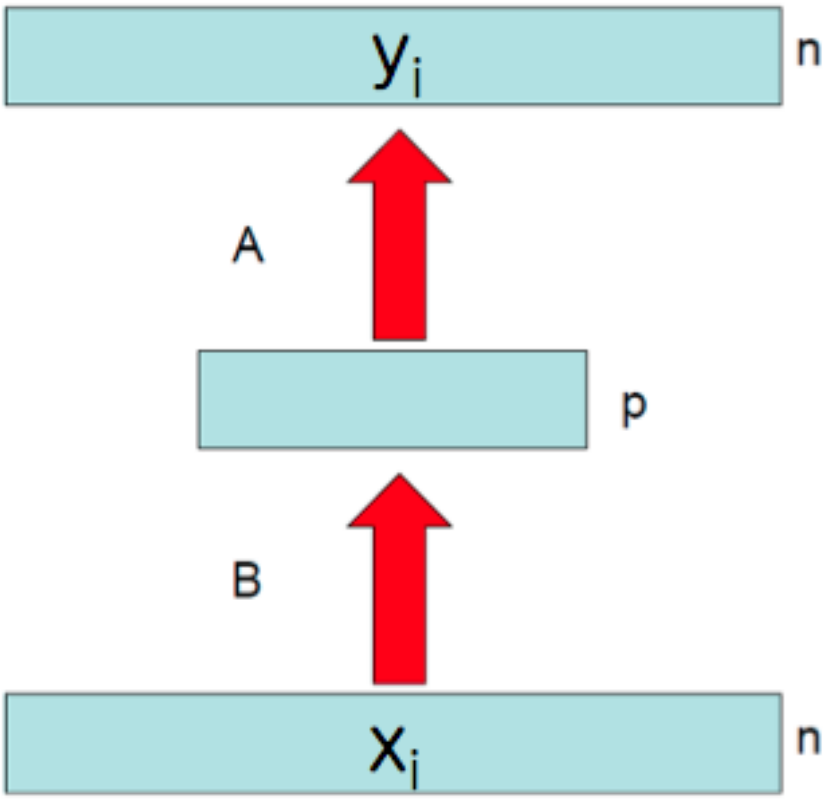
Why Deep Neural Networks?: **Unsupervised Feature Learning**

- **기계학습에 많은 학습 데이터 필요**
  - 소량의 학습 데이터
    - 학습 데이터 구축 비용/시간
  - **대량의 원시 코퍼스 (unlabeled data)**
    - **Semi-supervised**, **Unsupervised …**

- **Deep Neural Network**
  - **Pre-training 방법을 통해 대량의 원시 코퍼스에서 자질 학습**
  - Restricted Boltzmann Machines (RBM)
  - Stacked **Autoencoder**, Stacked Denosing Autoencoder
  - **Word Embedding (for NLP)**

Autoencoder

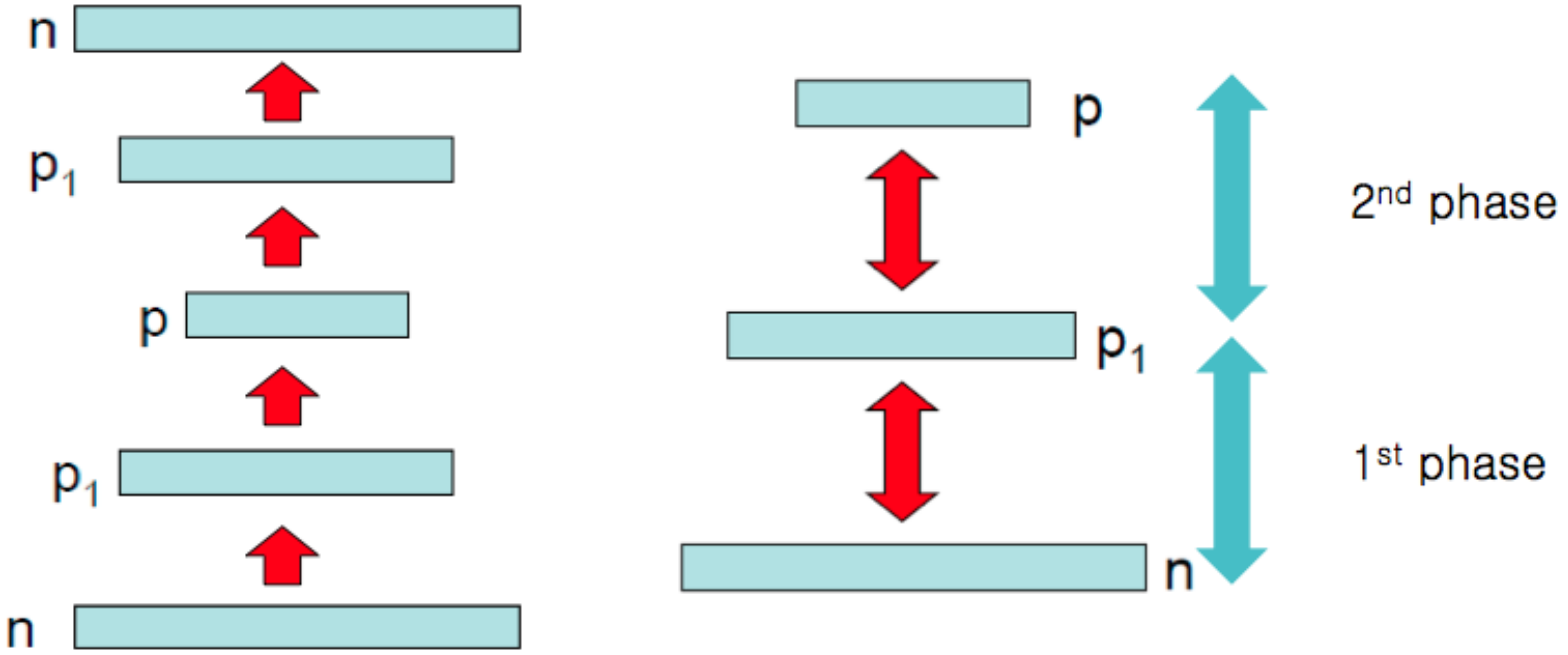
- **Autoencoder** is an **NN** whose **desired output is the same as the input**
  - To **learn a compressed representation (encoding)** for a set of data.
  - Find weight vectors A and B that minimize: $\Sigma_i(y_i-x_i)^2$



<겨울학교14 Deep Learning 자료 참고>

Stacked Autoencoders

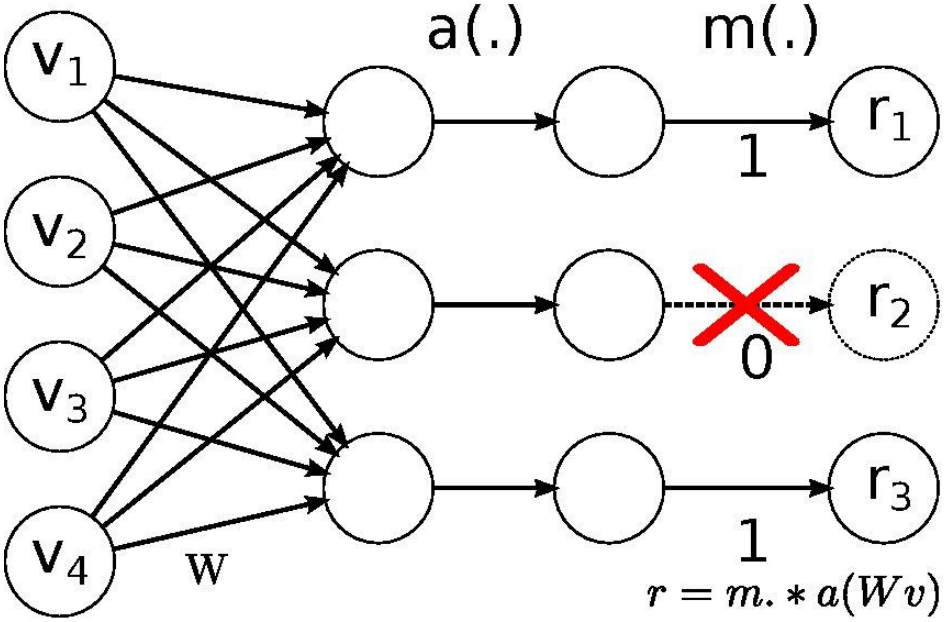
- After training, the hidden node extracts **features** from the input nodes
- Stacking autoencoders constructs a deep network



<겨울학교14 Deep Learning 자료 참고>

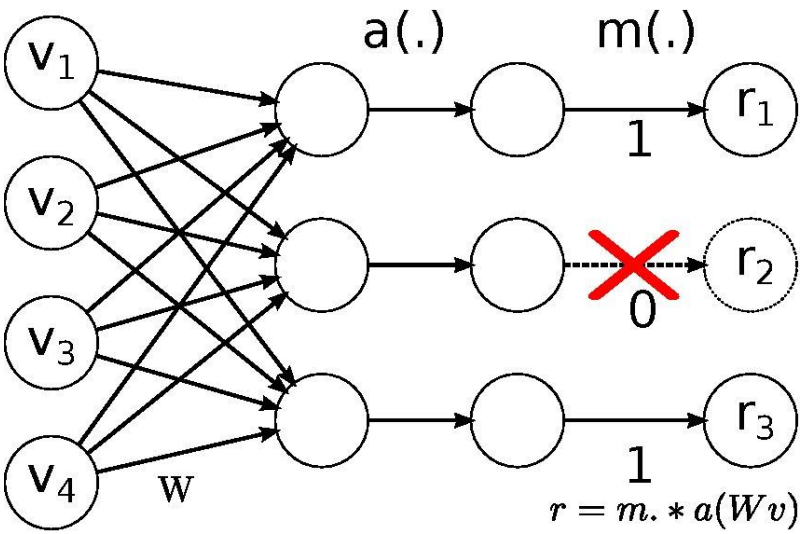
Dropout (Hinton12)

- In training, **randomly dropout** hidden units with probability p.

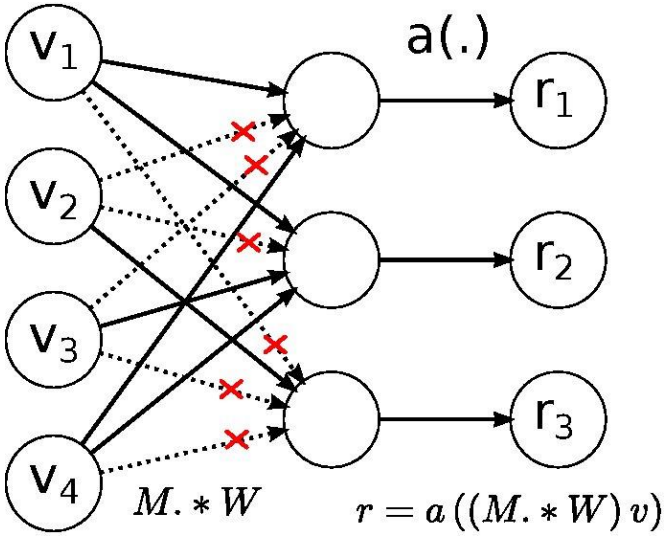


<겨울학교14 Deep Learning 자료 참고>

DropConnect (ICML13)



Dropout

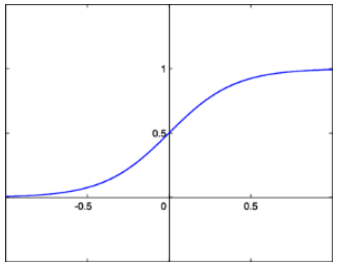


DropConnect

- DropConnect masks the weight.

Rectified Linear Hidden Unit (ReLU)

- **Sparse Coding**
  - Allow only a small number of computational units to have non -zero values

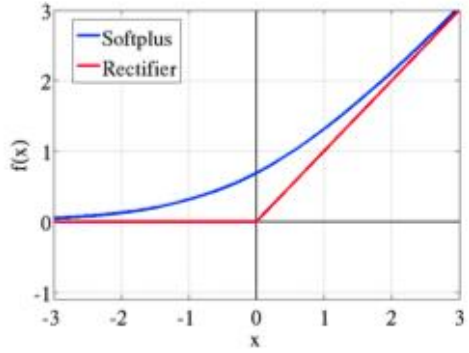


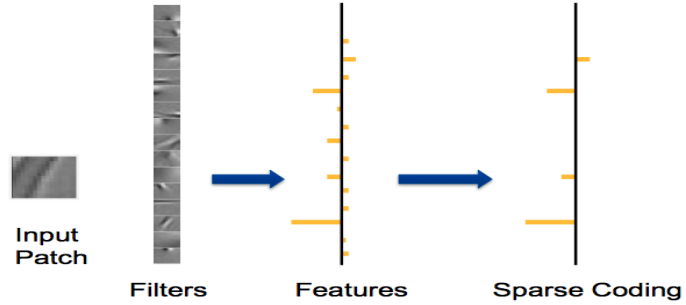
Sigmoid

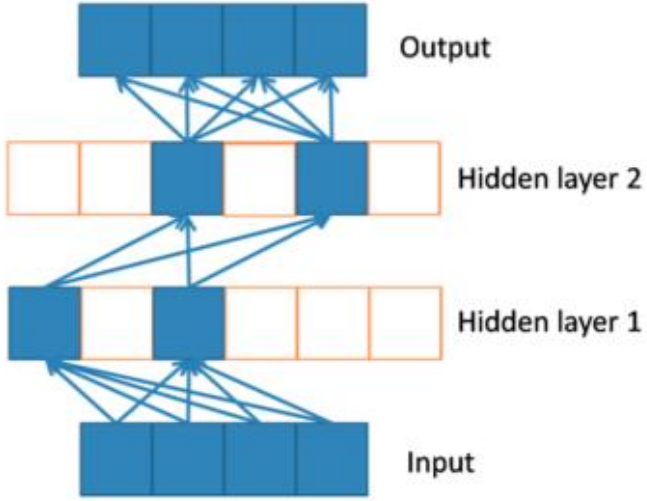
f(net) = 1/(1+exp(net))

ReLU

f(net) = max(0, net)







[Glorot11]

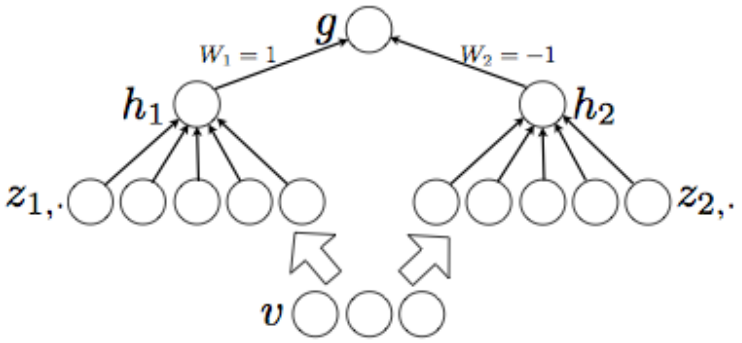
<겨울학교14 Deep Learning 자료 참고>

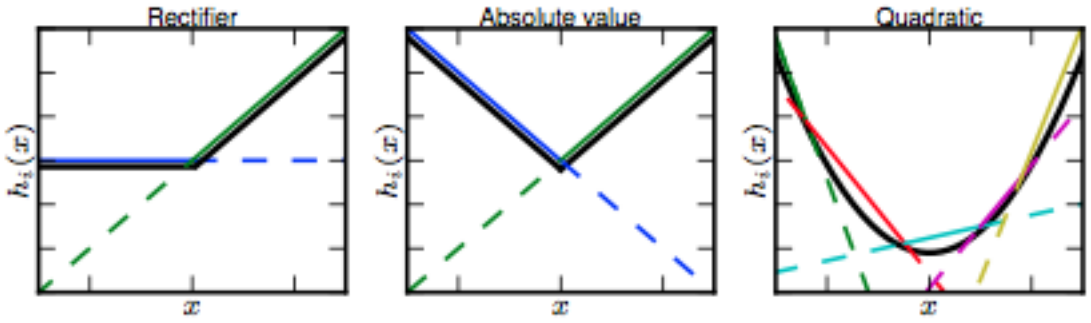
Maxout Networks (ICML13)

- **Learn the activation function**
- Maxout unit:
  - *k* is # linear models, *m* is # hidden units

$$h_i(x) = \max_{j \in [1,k]} z_{ij} \qquad z_{ij} = x^T W_{\ldots ij} + b_{ij}$$

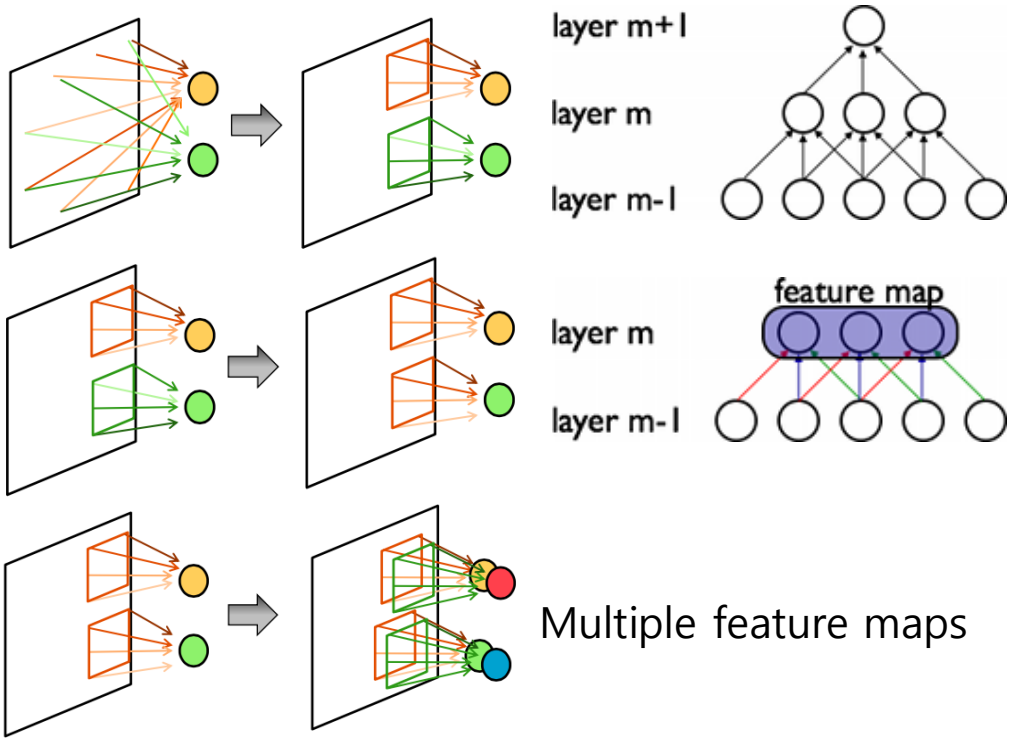
$$W \in R^{d \times m \times k} \quad \text{and} \quad b \in R^{m \times k}$$

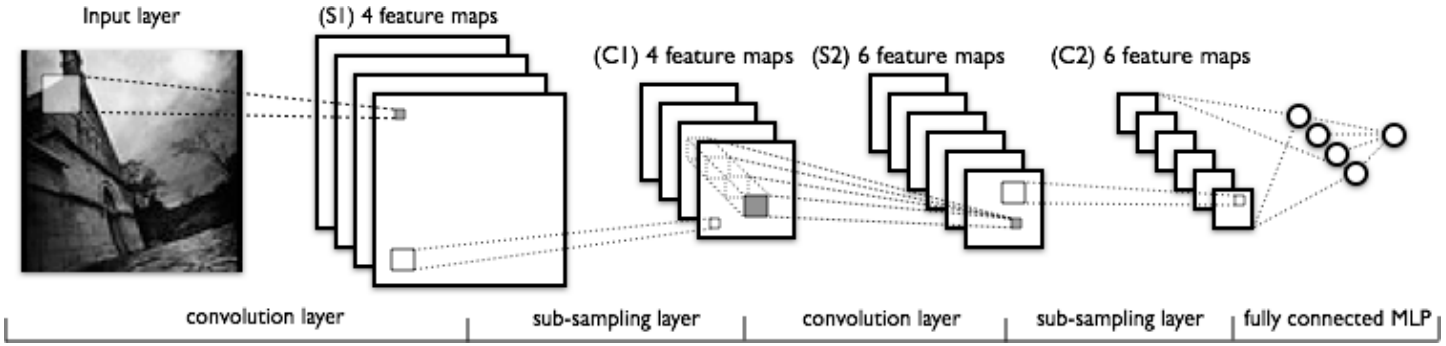




Convolutional Neural Network (LeCun98)

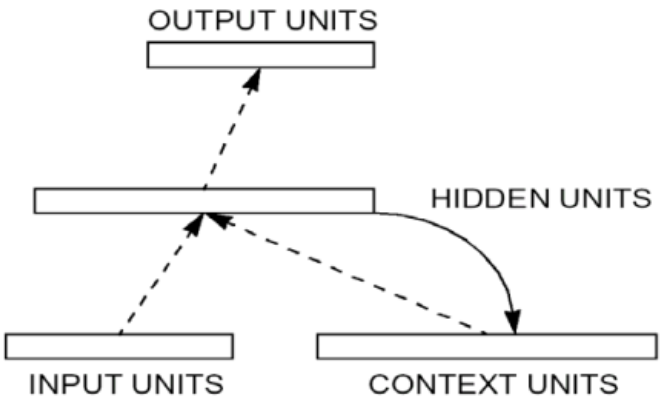
- Convolutional NN
  - Convolution Layer
    - Sparse Connectivity
    - Shared Weights
    - Multiple feature maps
  - Sub-sampling Layer
    - Average/max pooling
    - NxN→1
- Ex. LeNet

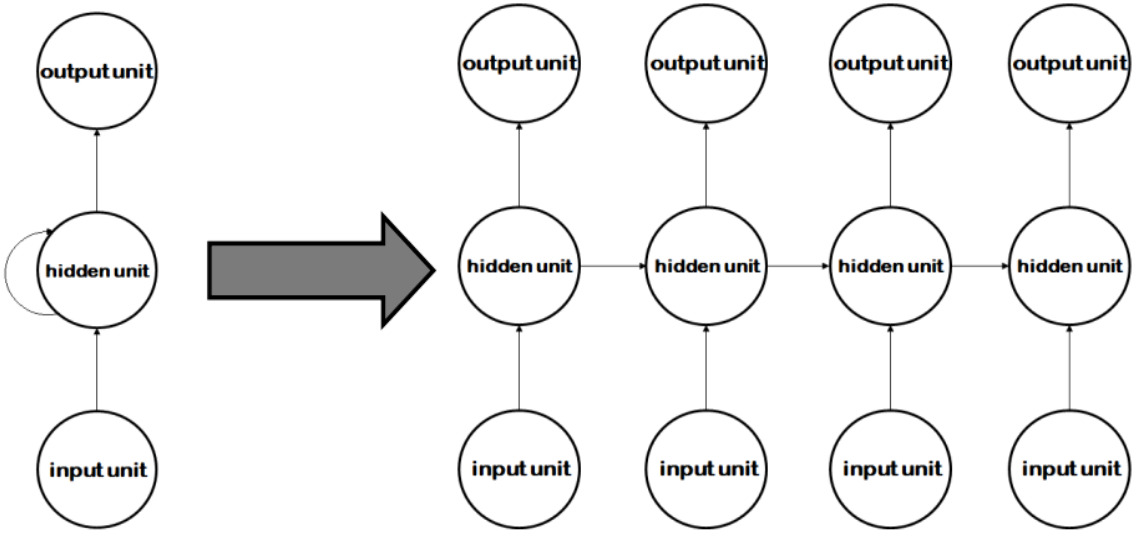




Recurrent Neural Network

- "Recurrent" property → dynamical system over time



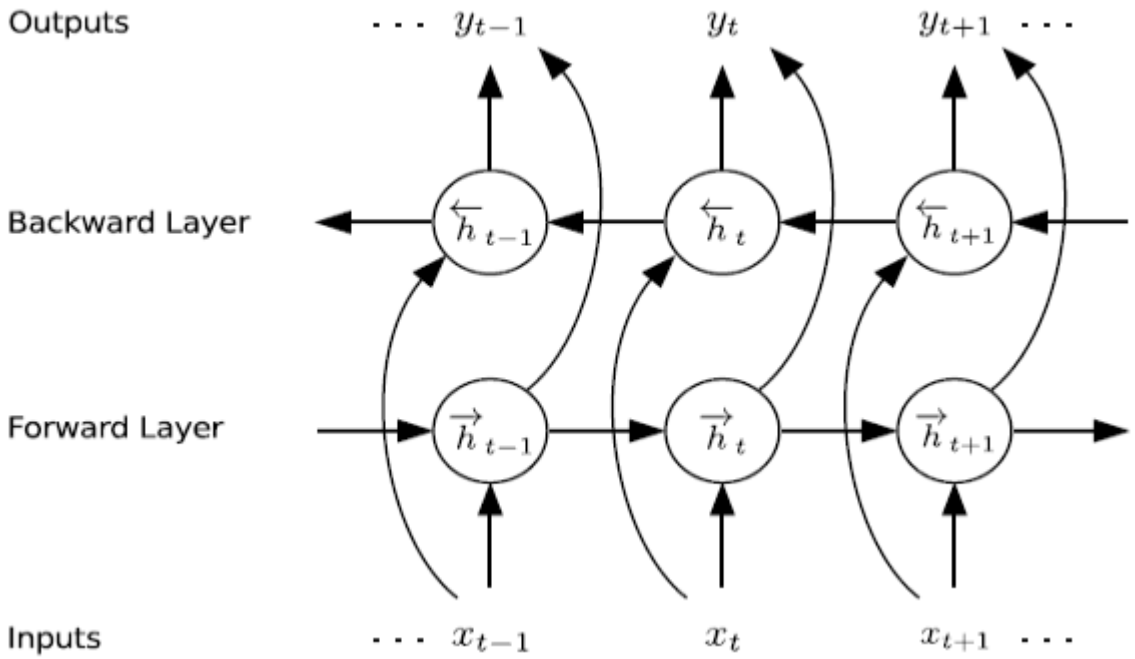


$$\boldsymbol{x} = (x_1, \ldots, x_T) \qquad h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

$$\boldsymbol{y} = (y_1, \ldots, y_T) \qquad y_t = W_{hy}h_t + b_y$$

Bidirectional RNN

- Exploit future context as well as past

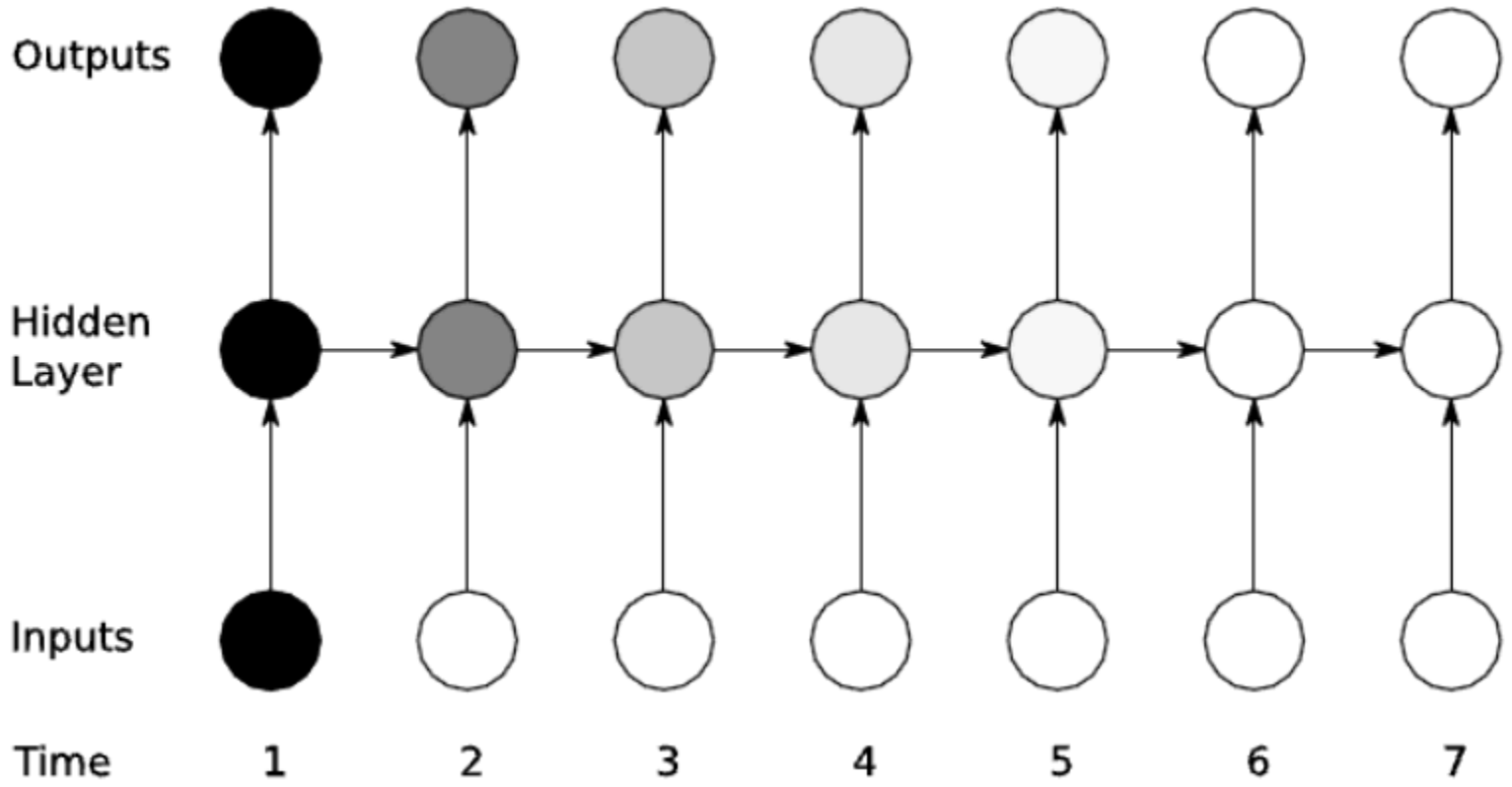


$$\overrightarrow{h_t} = H\left(W_{x\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}\right)$$

$$\overleftarrow{h_t} = H\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{t}}\right)$$

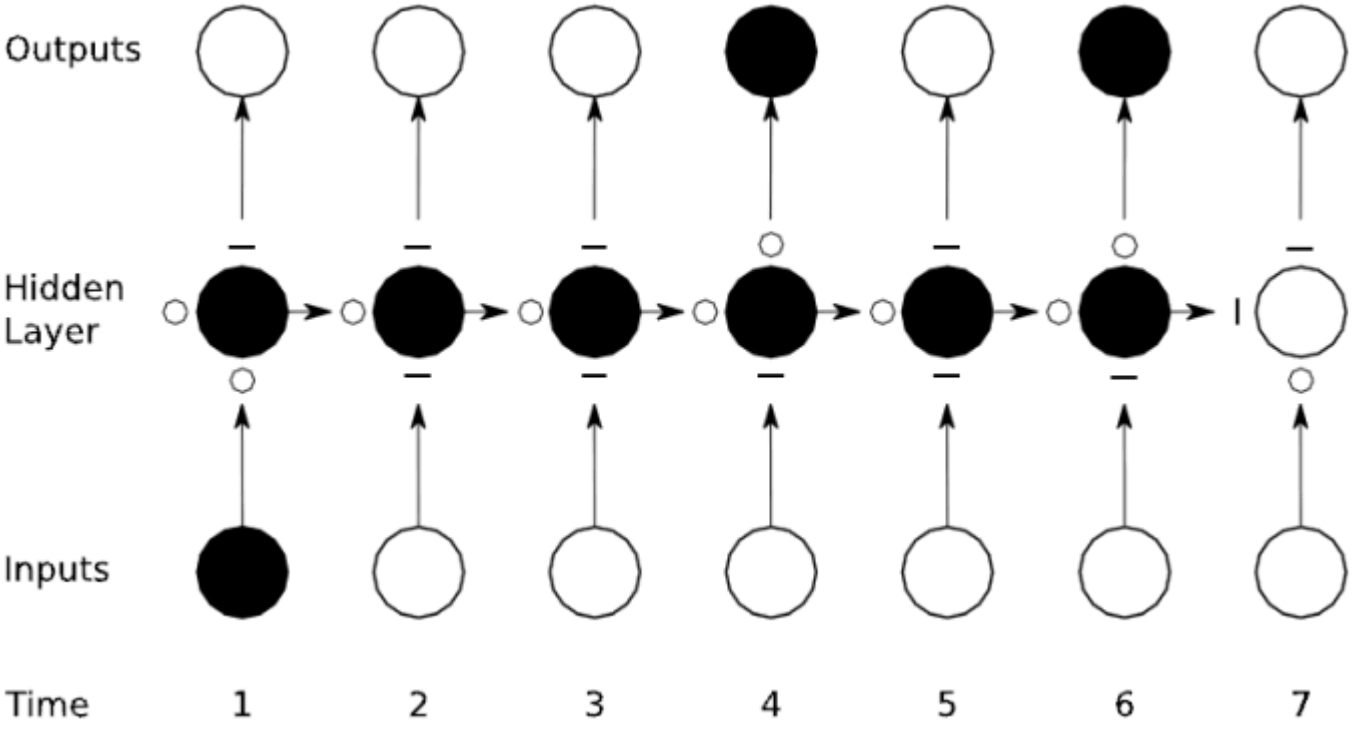
$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y$$

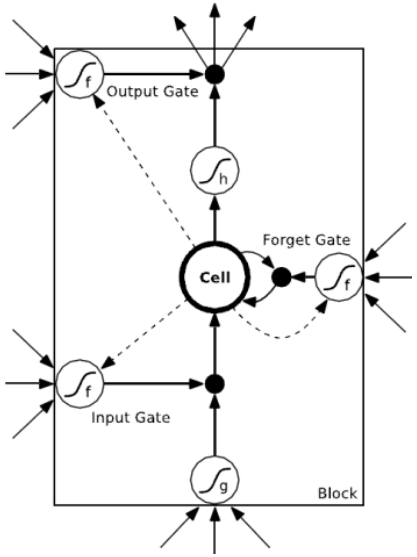
Vanishing Gradient Problem for RNN



Long Short-Term Memory RNN

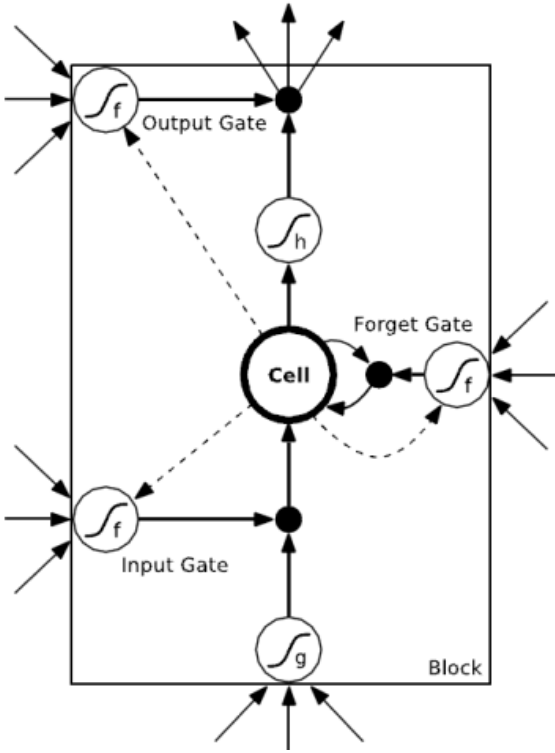
- LSTM can preserve gradient information



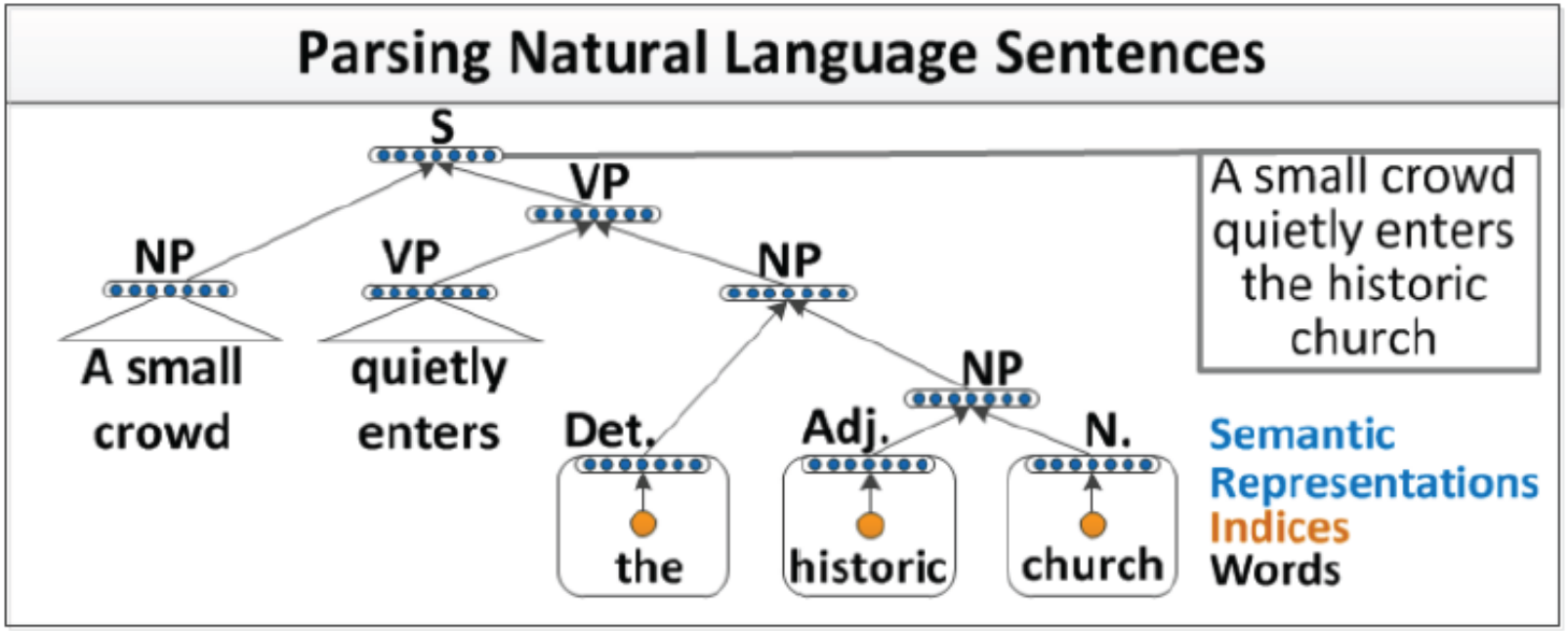


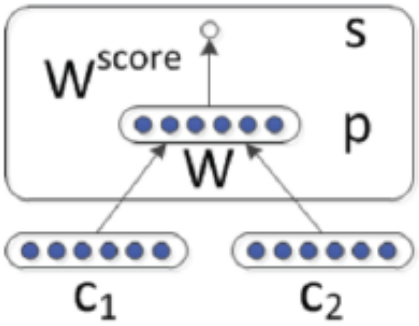
LSTM Block Architecture

- $i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$
- $f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right)$
- $c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$
- $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_0)$
- $h_t = o_t \tanh(c_t)$



Recursive Neural Network (ICML 13)





$$s = W^{score} p$$

$$p = f(W[c_1; c_2] + b)$$

차례

- 딥러닝 소개
- **Word Embedding**
- Word Embedding 응용
  - 딥러닝 기반의 자연어처리
- Phrase/Sentence Embedding

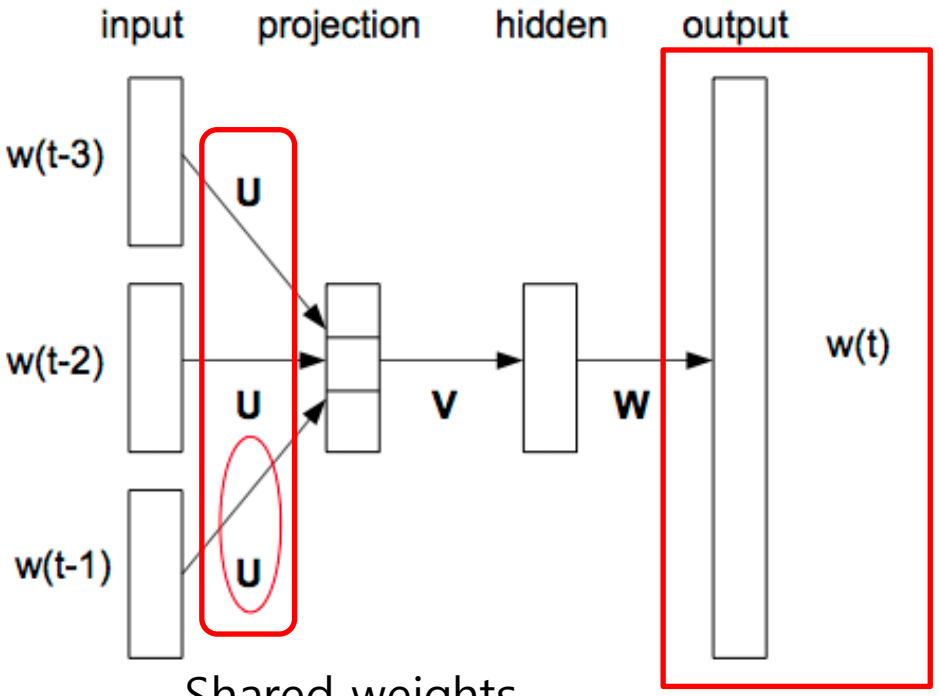
텍스트의 표현 방식

- **One-hot representation (or symbolic)**
  - Ex. [0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0]
  - **Dimensionality**
    - **50K (PTB) – 500K (big vocab) – 3M (Google 1T)**
  - **Problem**
    - **Motel [0 0 0 0 0 0 0 0 1 0 0] AND**
    - **Hotel [0 0 0 0 0 0 1 0 0 0 0] = 0**
- **Continuous representation**
  - Latent Semantic Analysis, Random projection
  - Latent Dirichlet Allocation, HMM clustering
  - **Neural Word Embedding**
    - **Dense vector**
    - By **adding supervision** from other tasks → **improve the representation**

Neural Network Language Model (Bengio00,03)

LT: |V|*d, Input(one hot): |V|*1 → $LT^T I$

- Idea
  - A **word and its context** is a **positive** training sample
  - A **random word in that same context → negative** training sample
  - Score(positive) > Score(neg.)
- Training complexity is high
  - **Hidden layer → output**
  - **Softmax in the output layer**
    - **Hierarchical softmax**
    - **Negative sampling**
    - **Ranking(hinge loss)**



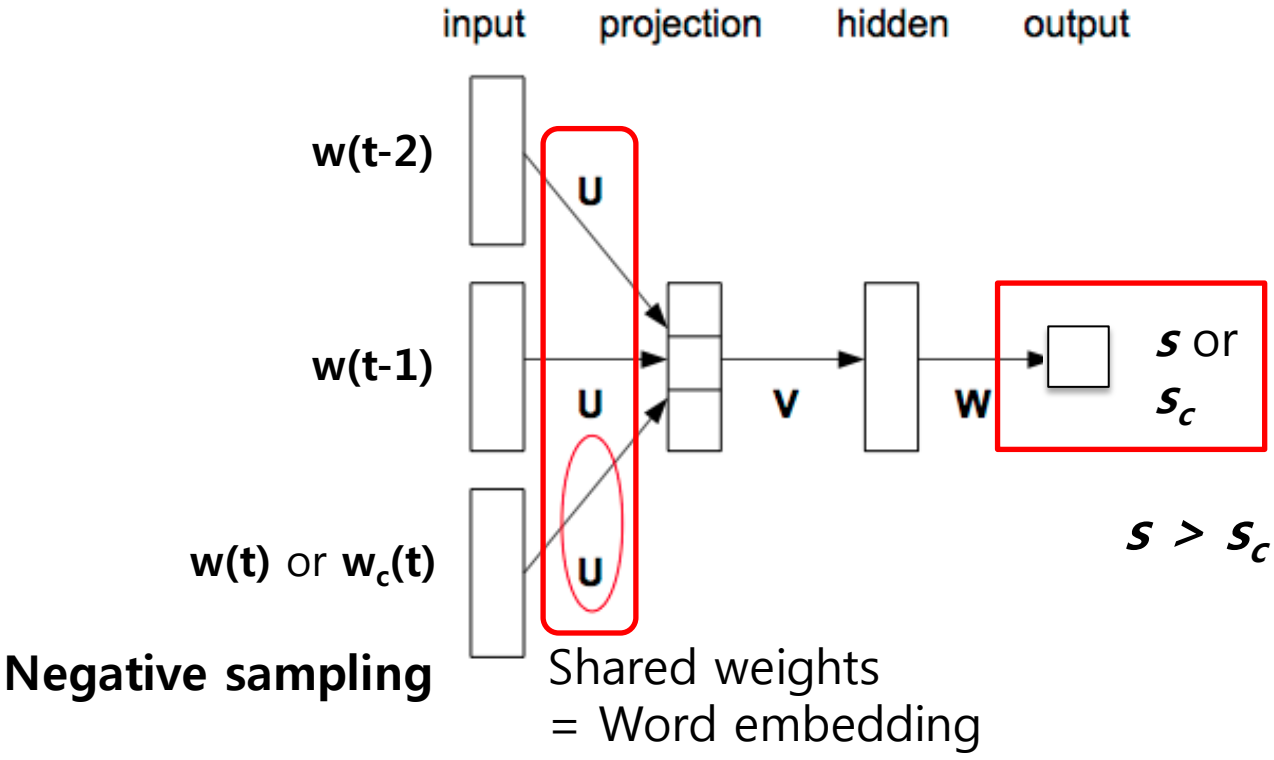
Shared weights
= Word embedding

Input	Dim: 1	Dim: 2	Dim: 3	Dim: 4	Dim: 5
1 (boy)	0.01	0.2	-0.04	0.05	-0.3
2 (girl)	0.02	0.22	-0.05	0.04	-0.4

Ranking-based Model (Collobert)

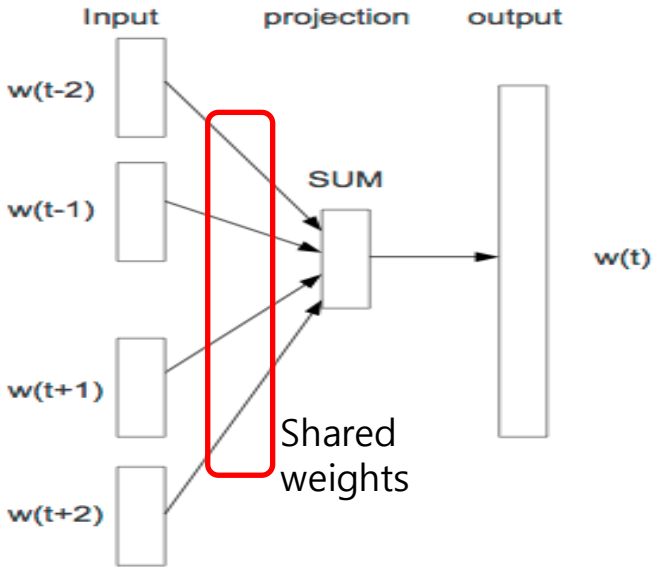
Ranking(hinge loss)[2,11]: $\max(0, 1 - s + s_c)$

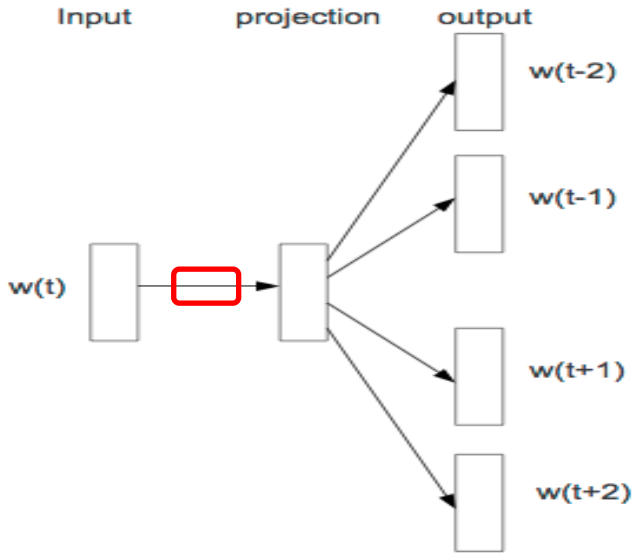
Ranking(logit loss): $\log(1 + \exp(s_c - s))$ → 본 연구 추가



Word2Vec: CBOW, Skip-Gram

- **Remove the hidden layer → Speedup 1000x**
  - **Negative sampling**
  - **Frequent word sampling**
  - **Multi-thread (no lock)**
- **Continuous Bag-of-words (CBOW)**
  - Predicts the current word given the con text
- **Skip-gram**
  - Predicts the surrounding words given the current word
  - **CBOW + DropOut/DropConnect**





Global Vector (EMNLP14)

- Capturing local co-occurrence statistics

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

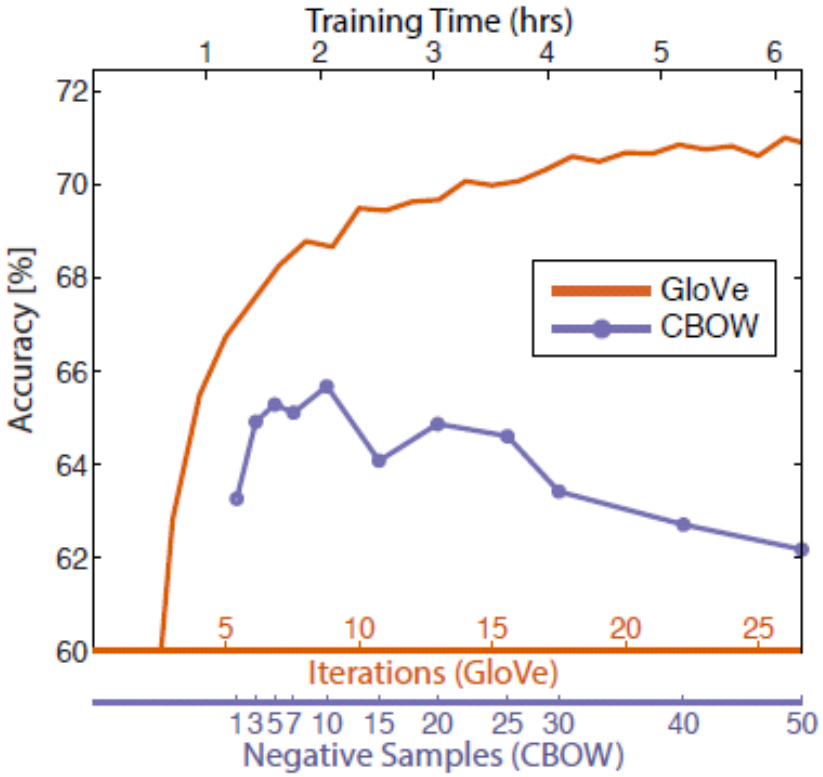
$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}} \qquad w_i^T \tilde{w}_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)\,.$$

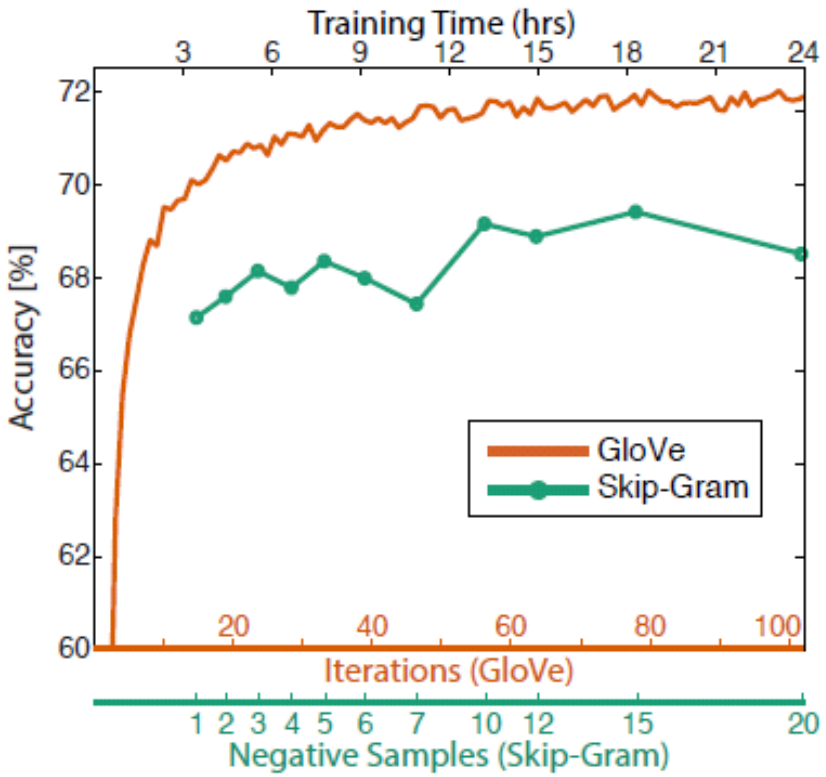
$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2 \qquad f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}.$$

Skip-Gram: $J = -\sum_{\substack{i \in \text{corpus} \\ j \in \text{context}(i)}} \log Q_{ij}\,. \qquad Q_{ij} = \frac{\exp(w_i^T \tilde{w}_j)}{\sum_{k=1}^{V} \exp(w_i^T \tilde{w}_k)}.$

- Efficient use of statistics:
  - Can train on (comparably) little data and gigantic data!

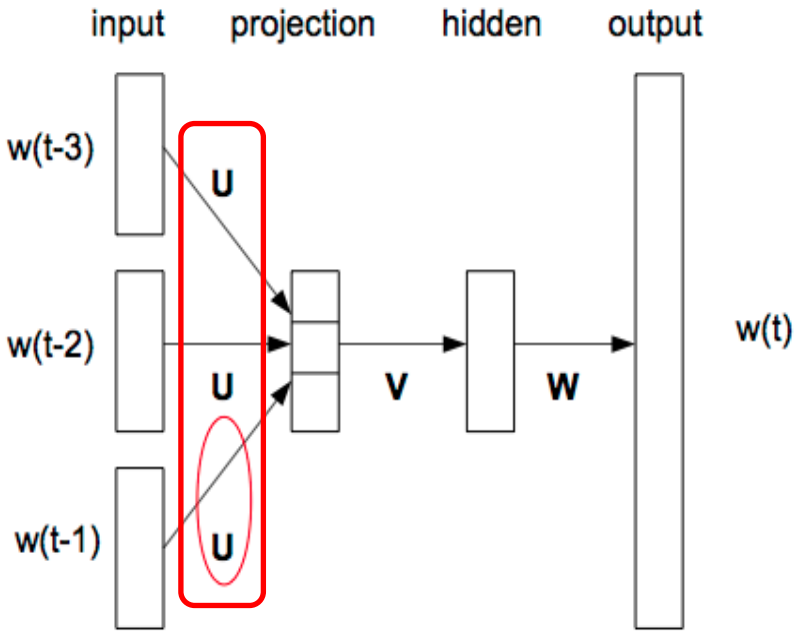
Global Vector (EMNLP14) – cont'd



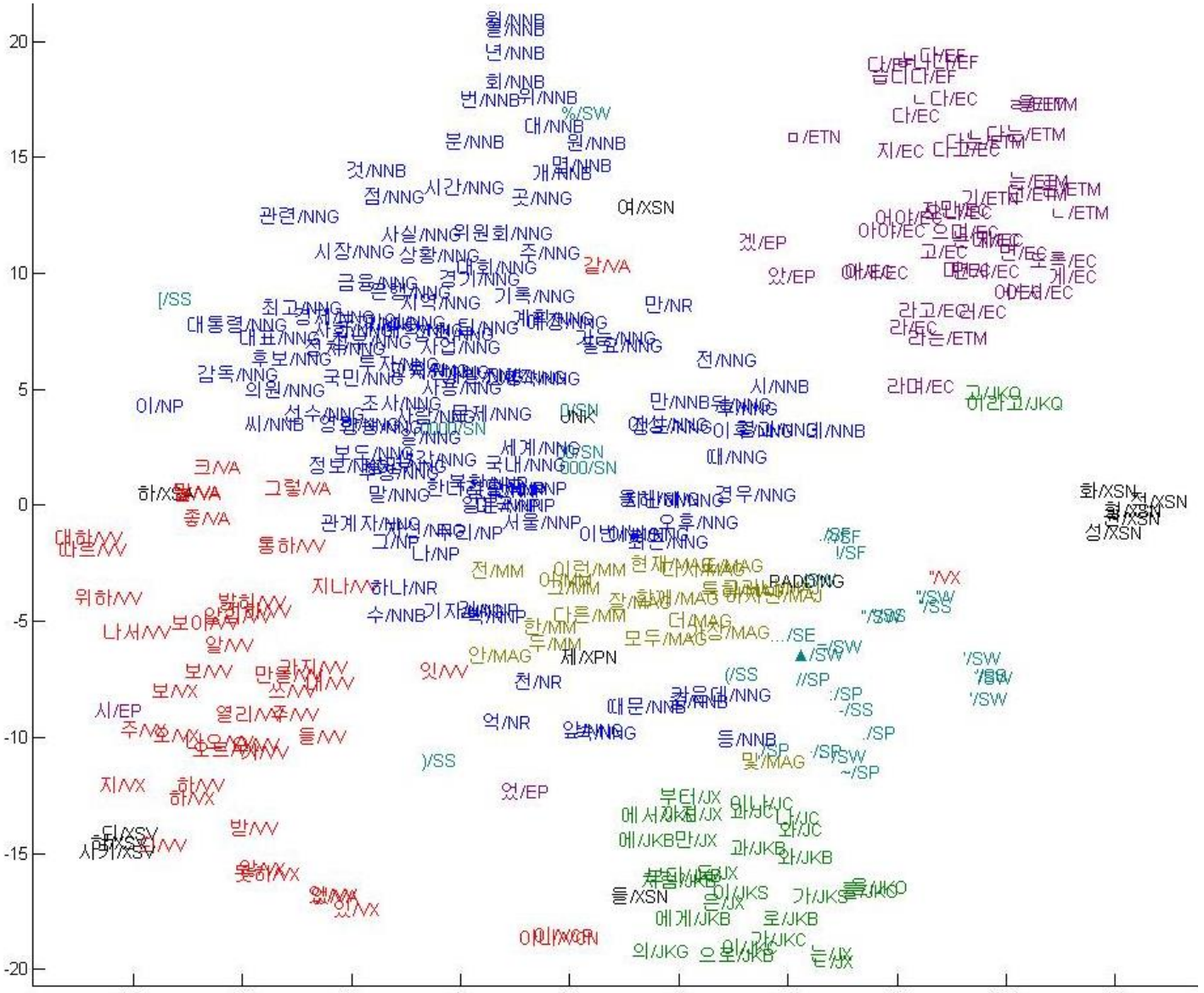


한국어 Word Embedding: NNLM

- Data
  - 세종코퍼스 원문 + Korean Wiki abstract + News data
    - **2억 8000만 형태소**
  - Vocab. size: **60,000**
    - 모든 형태소 대상 (기호, 숫자, 한자, 조사 포함)
    - **숫자 정규화 + 형태소/POS: 정부/NNG, 00/SN**
- **NNLM model**
  - Dimension: 50
  - **Matlab 구현**
  - **학습시간: 16일**



한국어 Word Embedding: NNLM



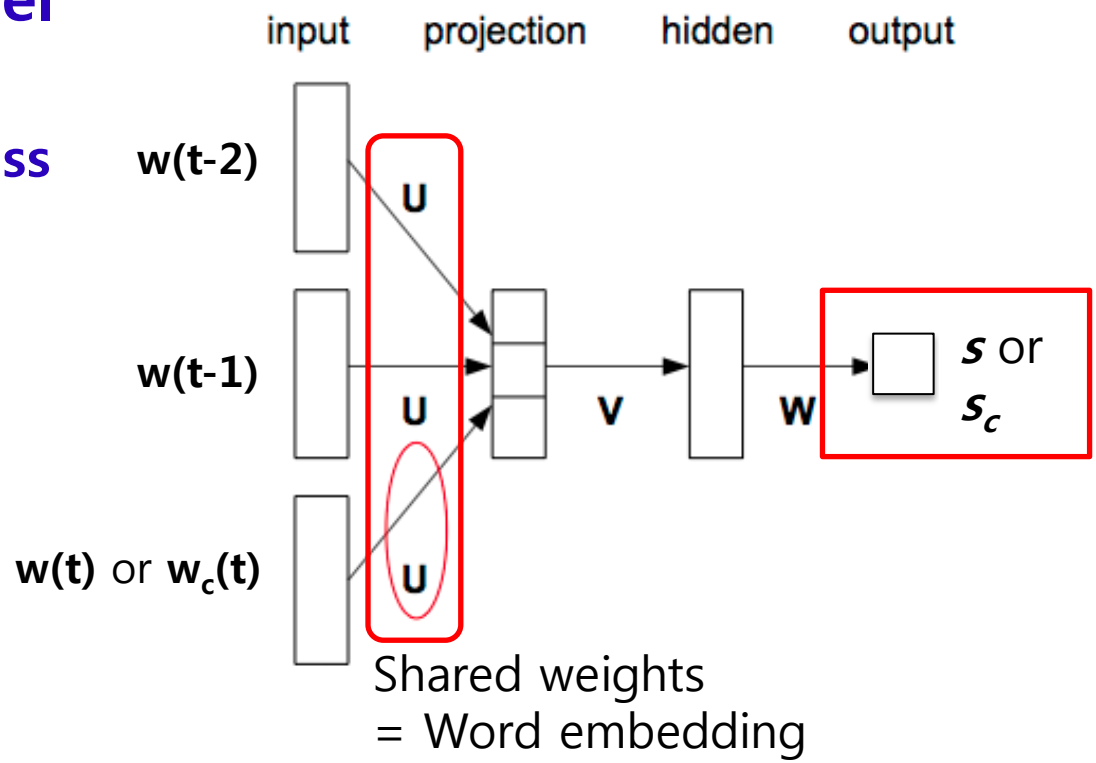
한국어 Word Embedding: Ranking-based

- Data (앞과 동일)
  - **2억 8000만 형태소**
  - Vocab. size: **60,000**
- **Ranking-based model**
  - 기존 연구: Hinge loss
  - **본 연구 추가: Logit loss**
  - Dimension: 50
  - Matlab 구현
  - **학습시간: 1.9일**

Ranking(hinge loss)[2,11]: $\max(0, 1 - s + s_c)$

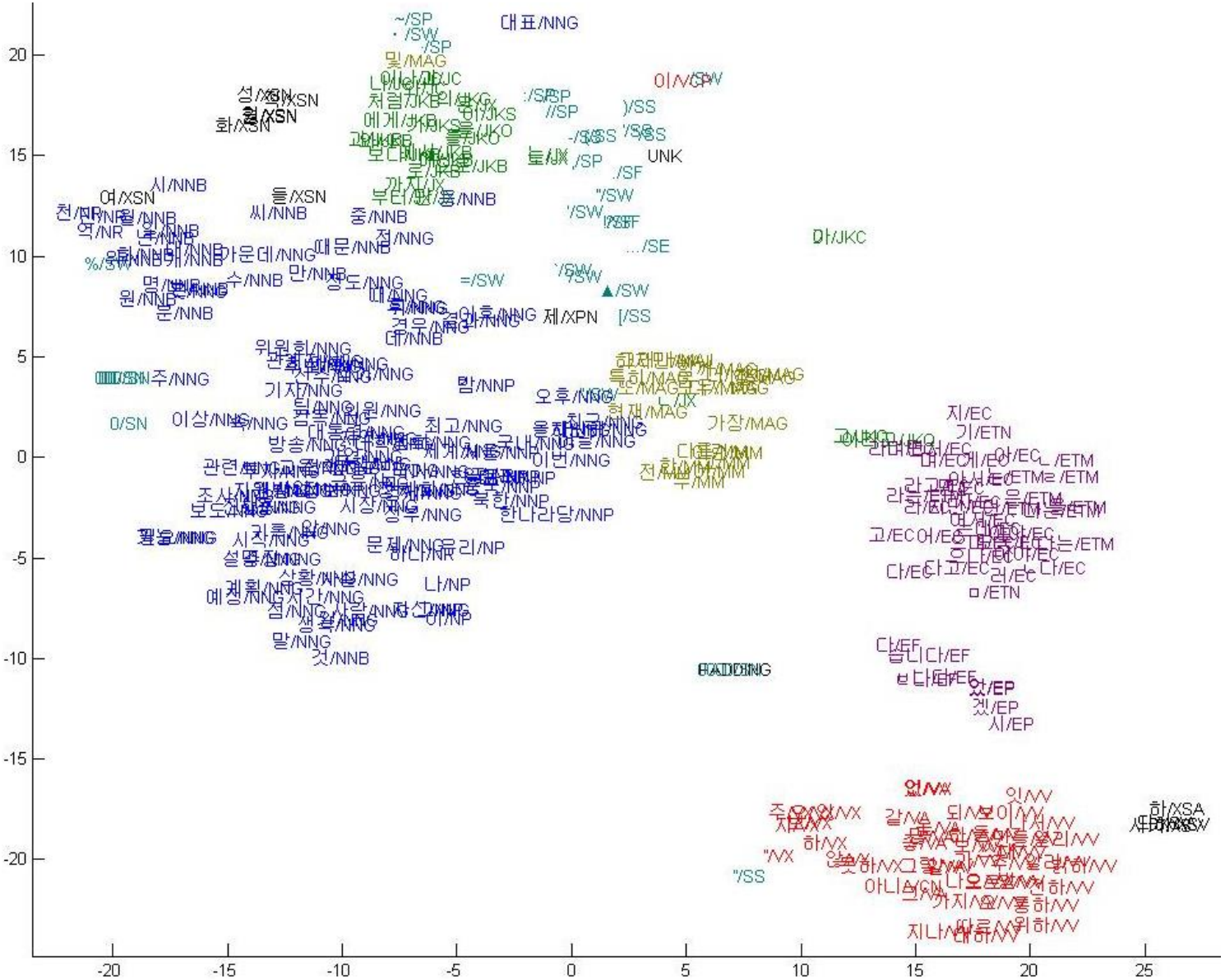
Ranking(logit loss): $\log(1 + \exp(s_c - s))$

→ 본 연구 추가



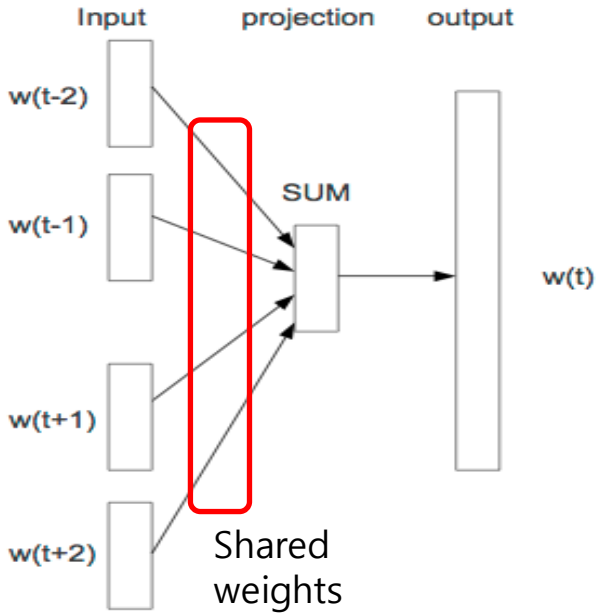
Shared weights
= Word embedding

한국어 Word Embedding: Ranking(Logit)

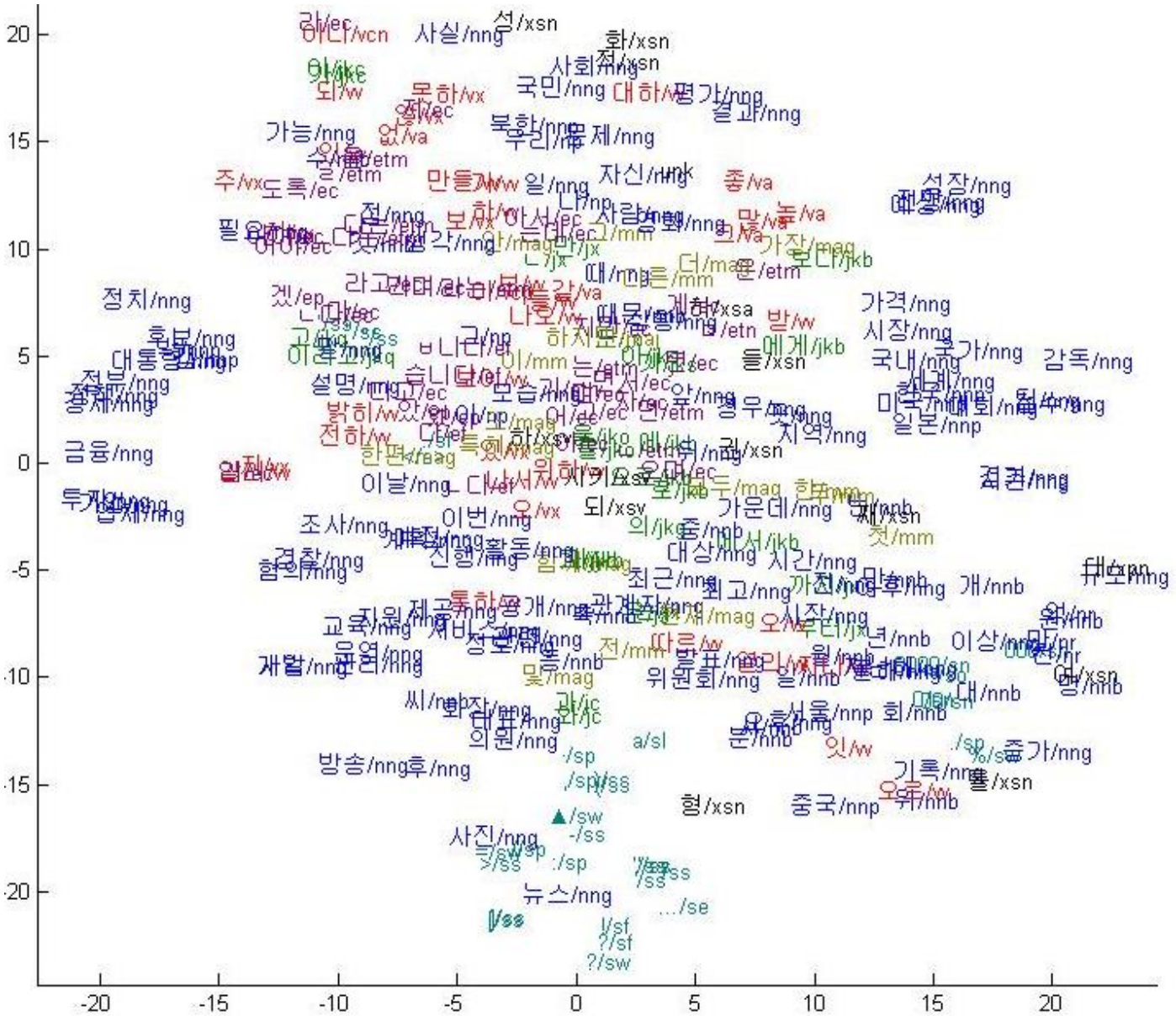


한국어 Word Embedding: Word2Vec(CBOW)

- Data
  - 2012-2013 News + Korean Wiki abstract:
    - **9GB raw text**
    - **29억 형태소**
  - Vocab. size: **100,000**
    - 모든 형태소 대상 (기호, 숫자, 한자, 조사 포함)
    - **숫자 정규화 + 영어 소문자 + 형태소/POS**
- **기존 Word2Vec(CBOW, Skip-Gram)**
  - 모델: **CBOW** > SKIP-Gram
  - **학습 시간: 24분**



한국어 Word Embedding: Word2Vec(CBOW)



차례

전이 기반의 한국어 의존구문분석: Forward

- **Transition-based(Arc-Eager): O(N)**

- 예: CJ그룹이$_1$ 대한통운$_2$ 인수계약을$_3$ 체결했다$_4$
  - **[root]**, **[CJ그룹이$_1$** 대한통운$_2$ …**]**, {}
    - 1: **Shift**
  - [root **CJ그룹이$_1$]**, **[대한통운$_2$** 인수계약을$_3$ …**]**, {}
    - 2: **Shift**
  - [root CJ그룹이$_1$ **대한통운$_2$]**, **[인수계약을$_3$** 체결했다$_4$**]**, {}
    - 3: **Left-arc(NP_MOD)**
  - [root **CJ그룹이$_1$]**, **[2←인수계약을$_3$** 체결했다$_4$**]**, {(**인수계약을$_3$→대한통운$_2$**)}
    - 4: **Shift**
  - [root CJ그룹이$_1$ **2←인수계약을$_3$]**, **[체결했다$_4$]**, {(인수계약을$_3$→대한통운$_2$)}
    - 5: **Left-arc(NP_OBJ)**
  - [root **CJ그룹이$_1$]**, **[3←체결했다$_4$]**, {(**체결했다$_4$→인수계약을$_3$**), …}
    - 6: **Left-arc(NP_SUB)**
  - **[root]**, **[(1,3)←체결했다$_4$]**, {(**체결했다$_4$→CJ그룹이$_1$**), …}
    - 7: **Right-arc(VP)**
  - [root→4 **(1,3)←체결했다$_4$]**, **[]**, {(**root→체결했다$_4$**), …}

Configuration: (S, B, A) [S = Stack, B = Buffer, A = Arcs]

Initial: $([\text{ROOT}], [w_1, \ldots, w_n], \{\ \})$

Terminal: $(S, [\ ], A)$

Shift:
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots, w_i, w_j], [\ldots], A)$

Reduce:
$([\ldots, w_i], B, A) \Longrightarrow ([\ldots], B, A)$

Right-Arc(l):
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots, w_i, w_j], [\ldots], A \cup \{(w_i, w_j, l)\})$

Left-Arc(l):
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots], [w_j, \ldots], A \cup \{(w_j, w_i, l)\})$

전이 기반의 한국어 의존구문분석: Backward

- **Transition-based(Arc-Eager) + Backward: O(N)**

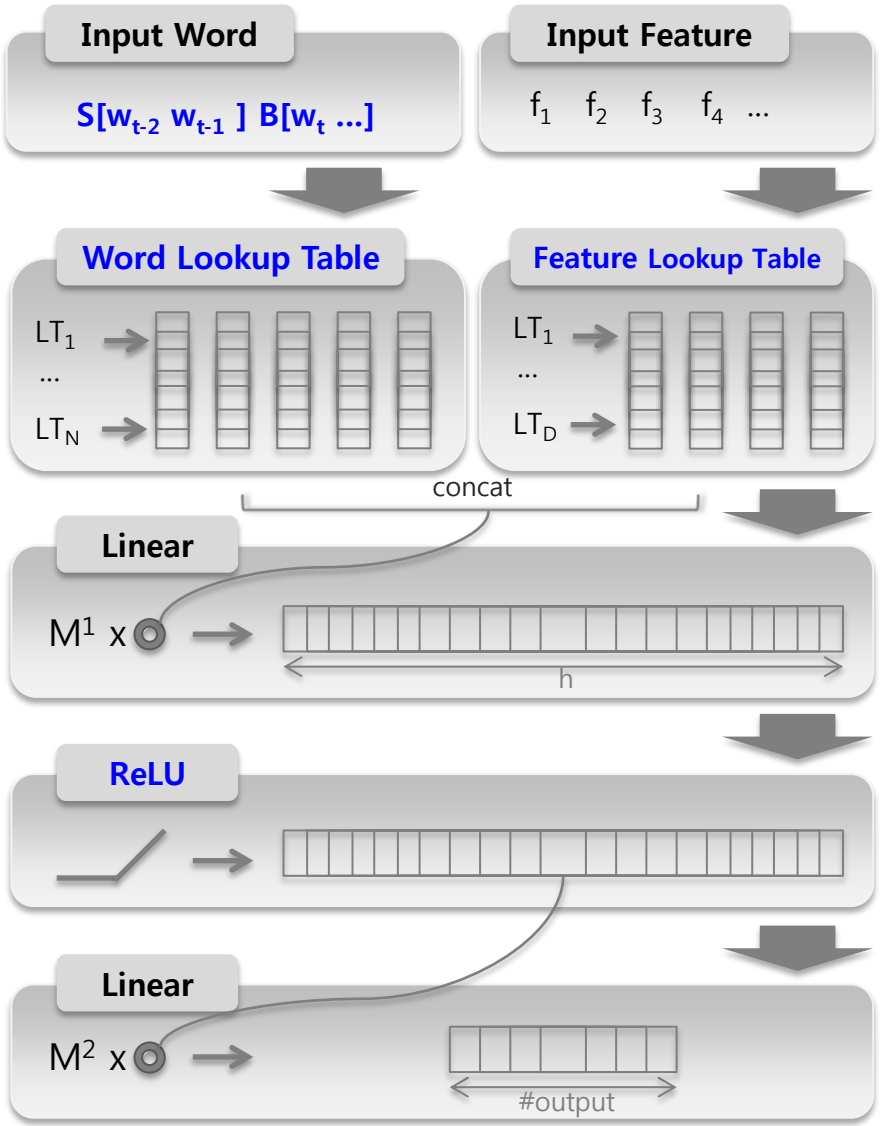
Configuration: (S, B, A) [S = Stack, B = Buffer, A = Arcs]
Initial: $([ROOT], [w_1, \ldots, w_n], \{\ \})$
Terminal: $(S, [\ ], A)$

Shift:
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots, w_i, w_j], [\ldots], A)$
Reduce:
$([\ldots, w_i], B, A) \Longrightarrow ([\ldots], B, A)$
Right-Arc(l):
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots, w_i, w_j], [\ldots], A \cup \{(w_i, w_j, l)\})$
Left-Arc(l):
$([\ldots, w_i], [w_j, \ldots], A) \Longrightarrow ([\ldots], [w_j, \ldots], A \cup \{(w_j, w_i, l)\})$

- 예: CJ그룹이 대한통운 인수계약을 체결했다
  - **[root]**, **[체결했다$_4$** 인수계약을$_3$ 대한통운$_2$ CJ그룹이$_1$**]**, {}
    - 1: **Right-arc(VP)**
  - [root→4 **체결했다$_4$]**, **[인수계약을$_3$** …**]**, {(**root→체결했다$_4$**)}
    - 2: **Right-arc(NP_OBJ)**
  - [root→4 체결했다$_4$→3 **인수계약을$_3$]**, **[대한통운$_2$** …**]**, {(**체결했다$_4$→인수계약을$_3$**), …}
    - 3: **Right-arc(NP_MOD)**
  - [root→4 체결했다$_4$→3 인수계약을$_3$→2 **대한통운$_2$]**, **[CJ그룹이$_1$]**, {(**인수계약을$_3$→대한통운$_2$**), …}
    - 4: **Reduce**
  - [root→4 체결했다$_4$→3 **인수계약을$_3$→2]**, **[CJ그룹이$_1$]**, {(인수계약을$_3$→대한통운$_2$), …}
    - 5: **Reduce**
  - [root→4 **체결했다$_4$→3]**, **[CJ그룹이$_1$]**, {(인수계약을$_3$→대한통운$_2$), …}
    - 6: **Right-arc(NP_SUB)**
  - [root→4 체결했다$_4$→(1,3) **CJ그룹이$_1$]**, **[]**, {(체결했다$_4$→CJ그룹이), …}

딥러닝 기반 한국어 의존구문분석 (한글 및 한국어14)

- **Transition-based + Backward**
  - **O(N)**
  - **세종코퍼스 → 의존 구문 변환**
    - 보조용언/의사보조용언 후처리
- **Deep Learning 기반**
  - **ReLU(> Sigmoid) + Dropout**
  - **Korean Word Embedding**
    - **NNLM, Ranking(hinge, logit)**
    - **Word2Vec**
  - **Feature Embedding**
    - **POS (stack + buffer)**
      - **자동 분석(오류 포함)**
    - **Dependency Label (stack)**
    - **Distance information**
    - **Valency information**
    - **Mutual Information**
      - **대용량 코퍼스 → 자동 구문 분석**

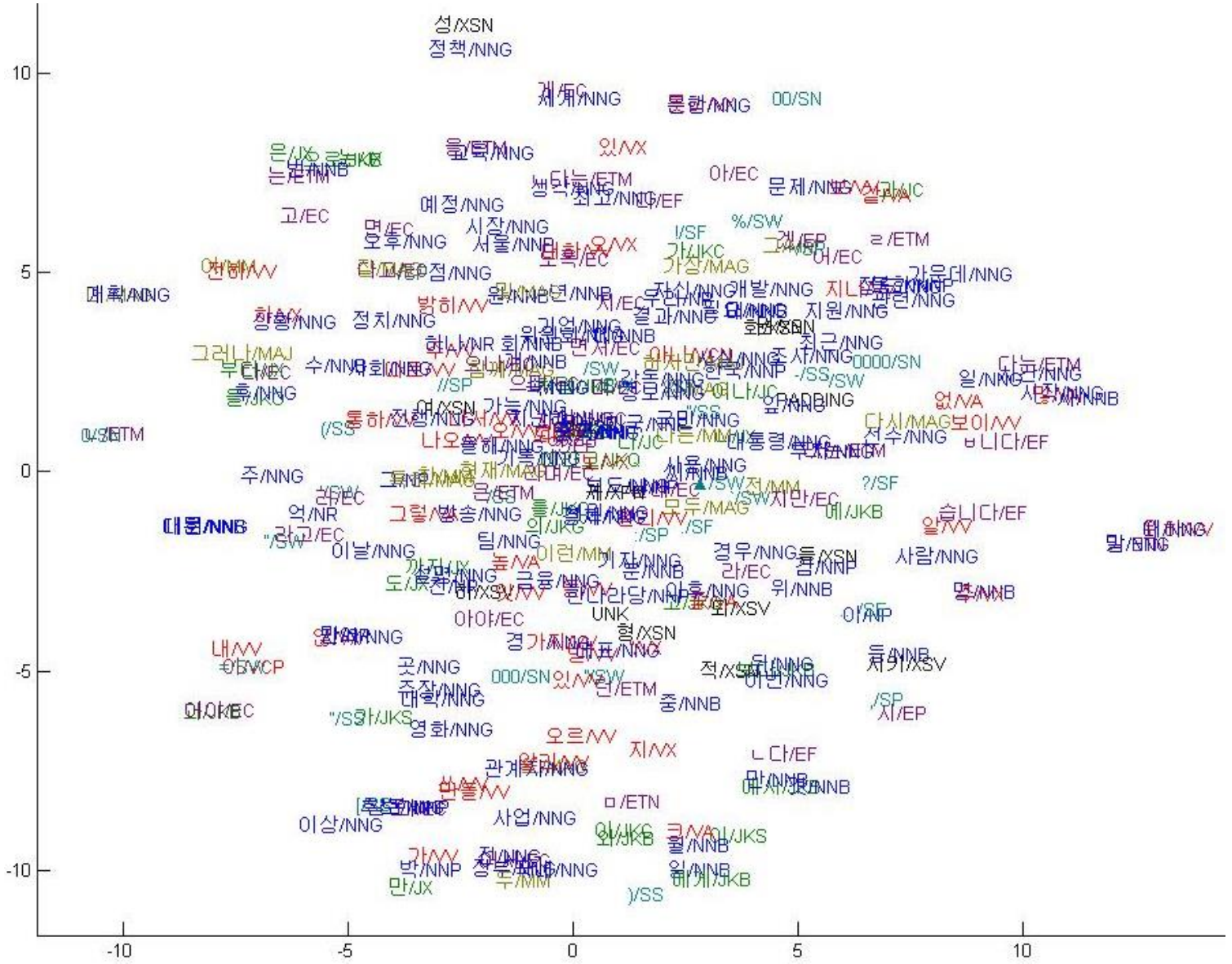


한국어 의존구문분석 실험 결과

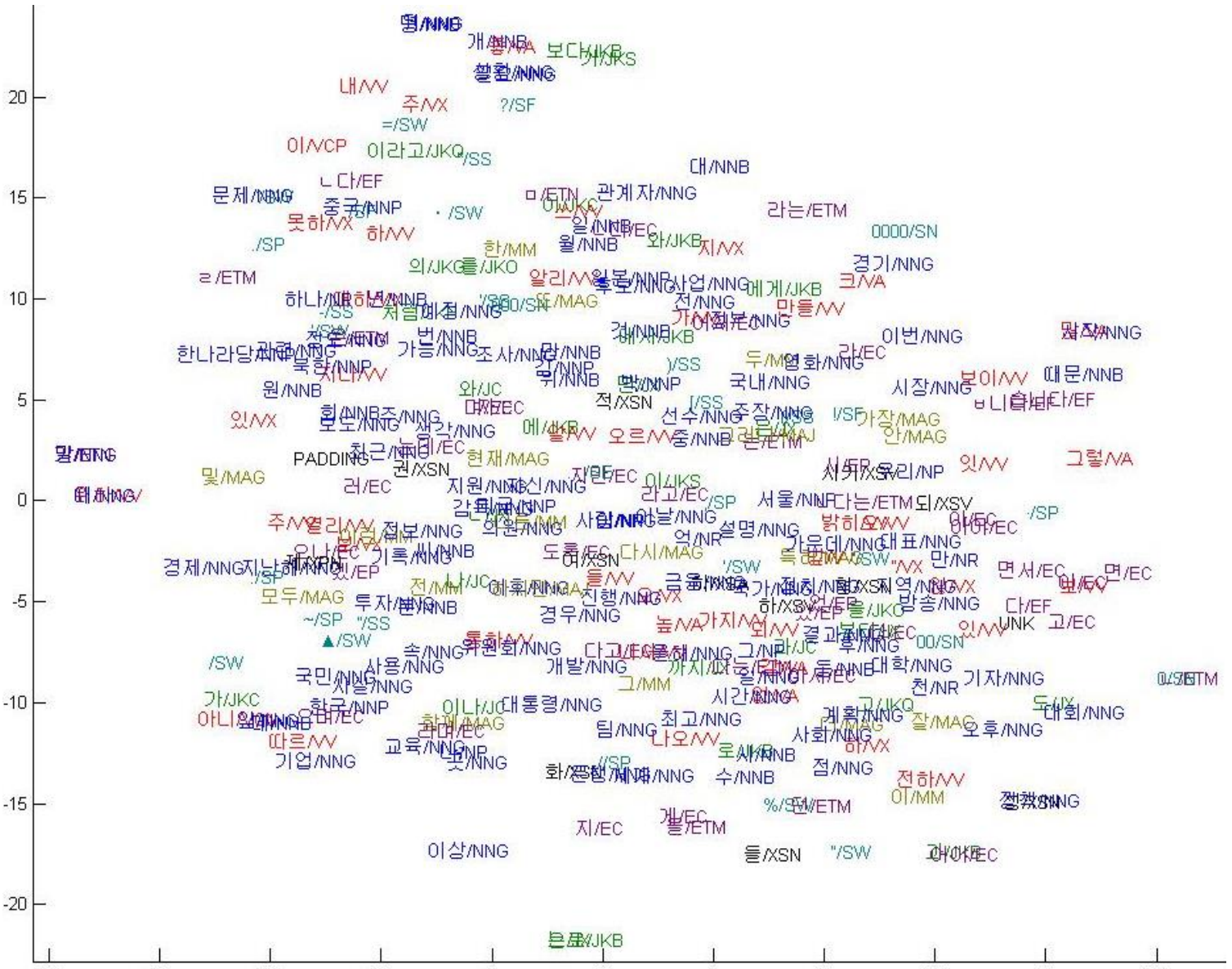
기존 의존 구문 분석	UAS	LAS
이용훈[15]: 국어정보베이스	88.42	-
오진영[16]: 세종코퍼스	87.03	-
임수종[17]: 세종코퍼스	88.15	-
J.D. Choi[18]: 세종코퍼스	85.47	83.47
박정열[19]: 세종코퍼스	86.43	-
안광모[20]: 세종코퍼스	87.52	-
딥 러닝+전이기반 의존 구문 분석 (세종코퍼스, 자동 분석 형태소 이용)	**UAS**	**LAS**
ReLU+dropout	89.56	87.35
NNLM+ReLU+dropout	90.05	87.87
NNLM+ReLU+MI feat.	89.91	87.58
NNLM+ReLU+dropout+MI feat.	**90.37**	**88.17**
NNLM+sigmoid+MI feat.	89.94	87.64
NNLM+sigmoid+dropout+MI feat.	90.27	88.03
Ranking(hinge loss)+ReLU+dropout +MI feat.	90.19	88.01
Ranking(logit loss)+ReLU+dropout +MI feat.	90.31	88.11
Word2vec+ReLU+dropout+MI feat.	90.27	87.97

- **기존연구: UAS 85~88%**
- **Structural SVM 기반 성능:**
  - UAS=89.99%
  - LAS=87.74%
- **Pre-training > no Pre.**
- **Dropout > no Dropout**
- **ReLU > Sigmoid**
- **MI feat. > no MI feat.**
- **Word Embedding 성능 순위**
  1. NNLM
  2. Ranking(logit loss)
  3. Word2vec
  4. Ranking(hinge loss)

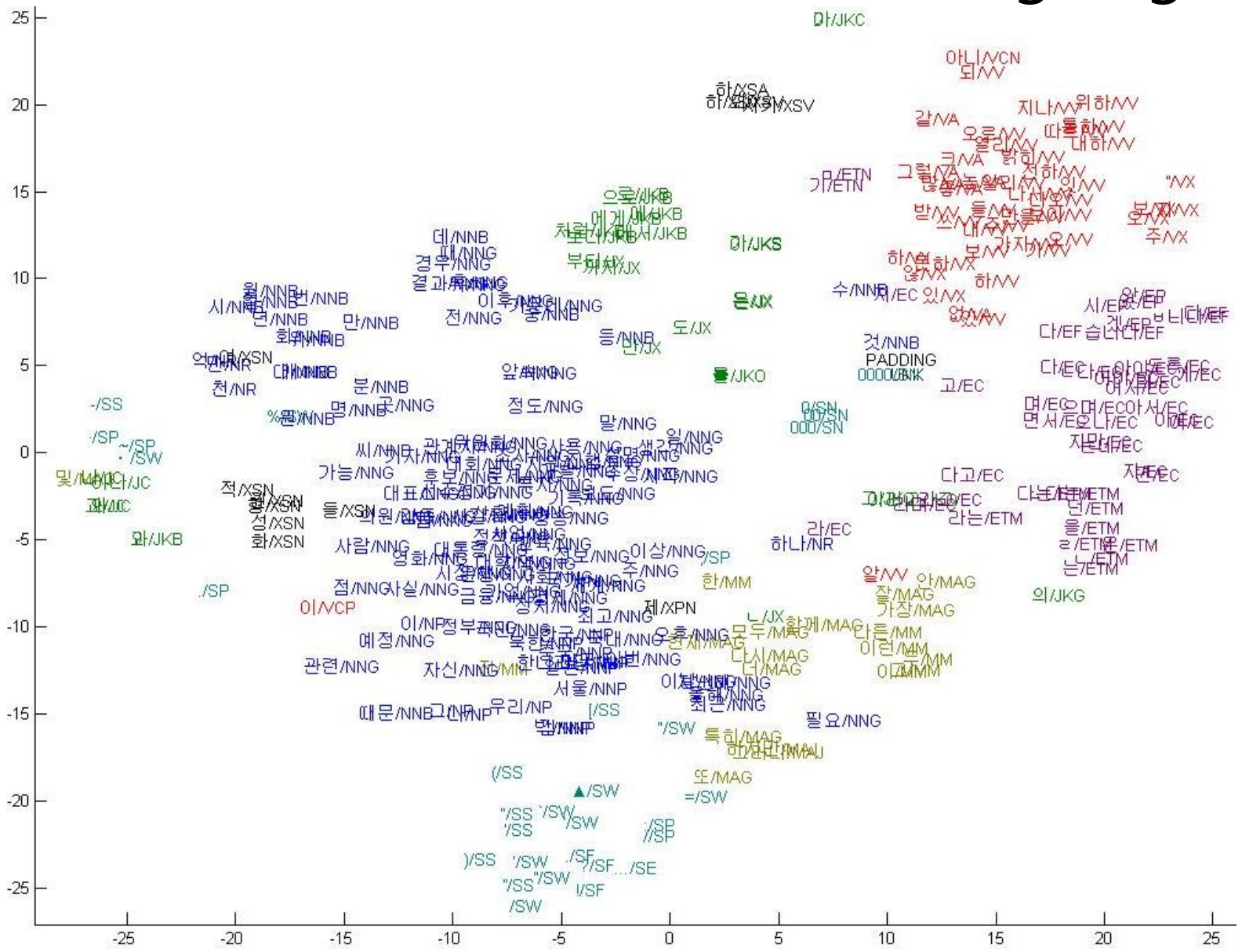
의존구문분석 학습 후 WE: No Pre-training



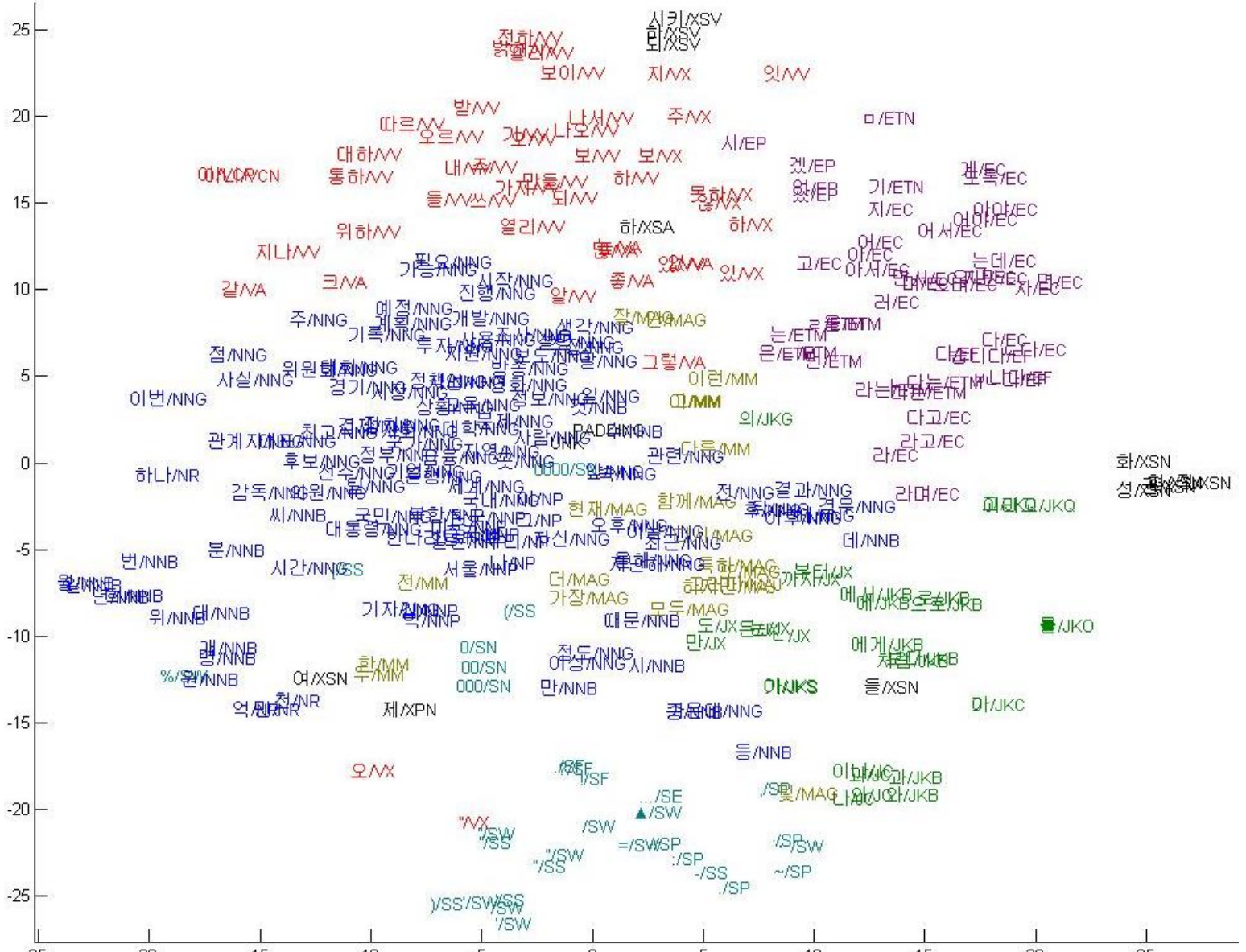
의존구문분석 학습 후 WE: Word2Vec



의존구문분석 학습 후 WE: Ranking(Logit)

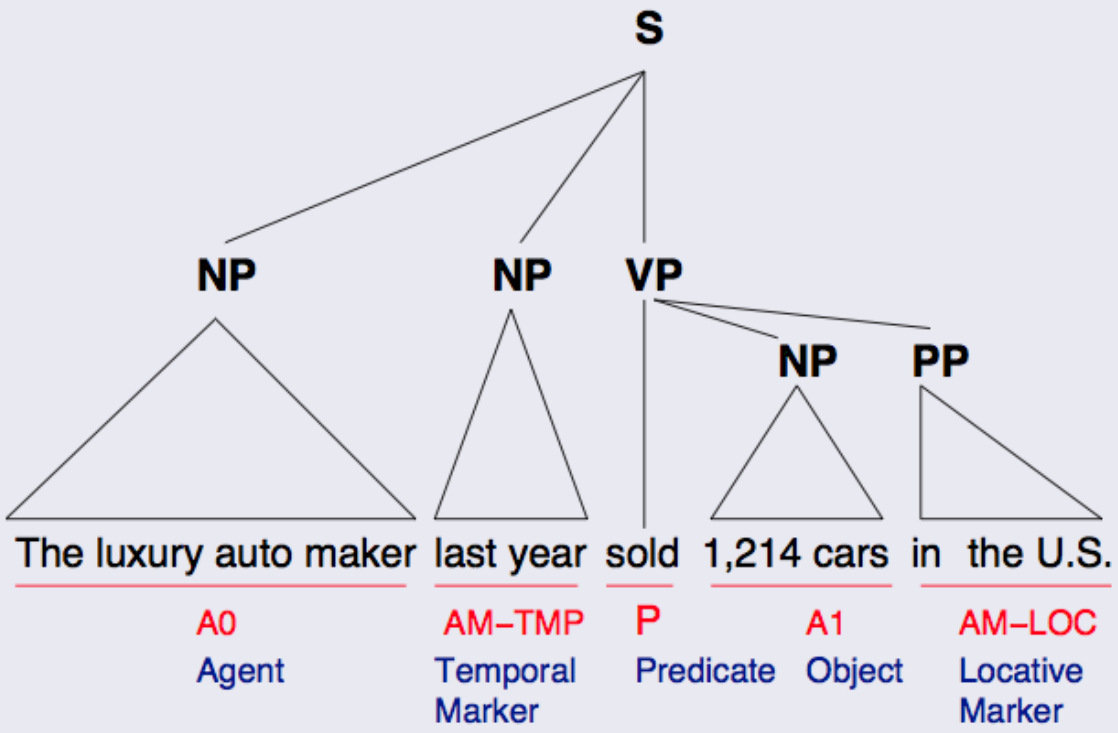


의존구문분석 학습 후 WE: NNLM



의미역 결정 (Semantic Role Labeling)

SRL $\stackrel{def}{=}$ detecting basic event structures such as *who* did *what* to *whom*, *when* and *where* [**IE** point of view]



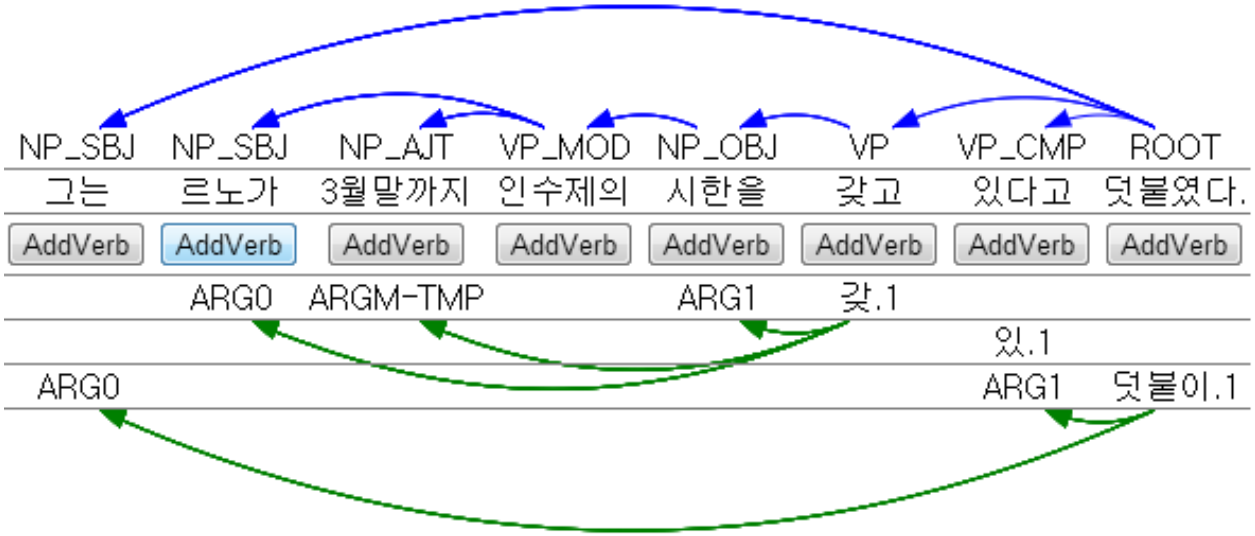
Korean PropBank

- Virginia Corpus
  - 군대 도메인, 54,500 단어
    → 도메인이 맞지 않음
- **Newswire Corpus**
  - 신문 도메인 (1994/6/2~2000/3/20), 131,800 단어
  - 구구조 기반 태깅을 의존 구조로 변환 → 4,882 문장
- **Frame file**
  - 2,749 xml files
    - 용언의 root (not stem)
    - 용언화 가능 명사 (deverbal noun)

- Frame file 예제
  - **덧붙.01**
  - English Define = add
  - Role set
    - ARG0: adder
    - ARG1: thing added
    - ARG2: added to
  - Mapping1
    - Rel = 덧붙이다
    - Src = sbj, Trg = ARG0
    - Src = obj, Trg = ARG1
    - Src = comp, Trg=ARG2

한국어 의미역 결정 (SRL)

- **서술어 인식(PIC)**
  - 그는 르노가 3월말까지 인수제의 시한을 [갖고]$_{갖.1}$ 있다고 [덧붙였다]$_{덧붙.1}$
- **논항 인식(AIC)**
  - 그는 [르노가]$_{ARG0}$ [3월말까지]$_{ARGM-TMP}$ 인수제의 [시한을]$_{ARG1}$ **[갖고]$_{갖.1}$** [있다고]$_{AUX}$ 덧붙였다
  - [그는]$_{ARG0}$ 르노가 3월말까지 인수제의 시한을 갖고 [있다고]$_{ARG1}$ **[덧붙였다]$_{덧붙.1}$**

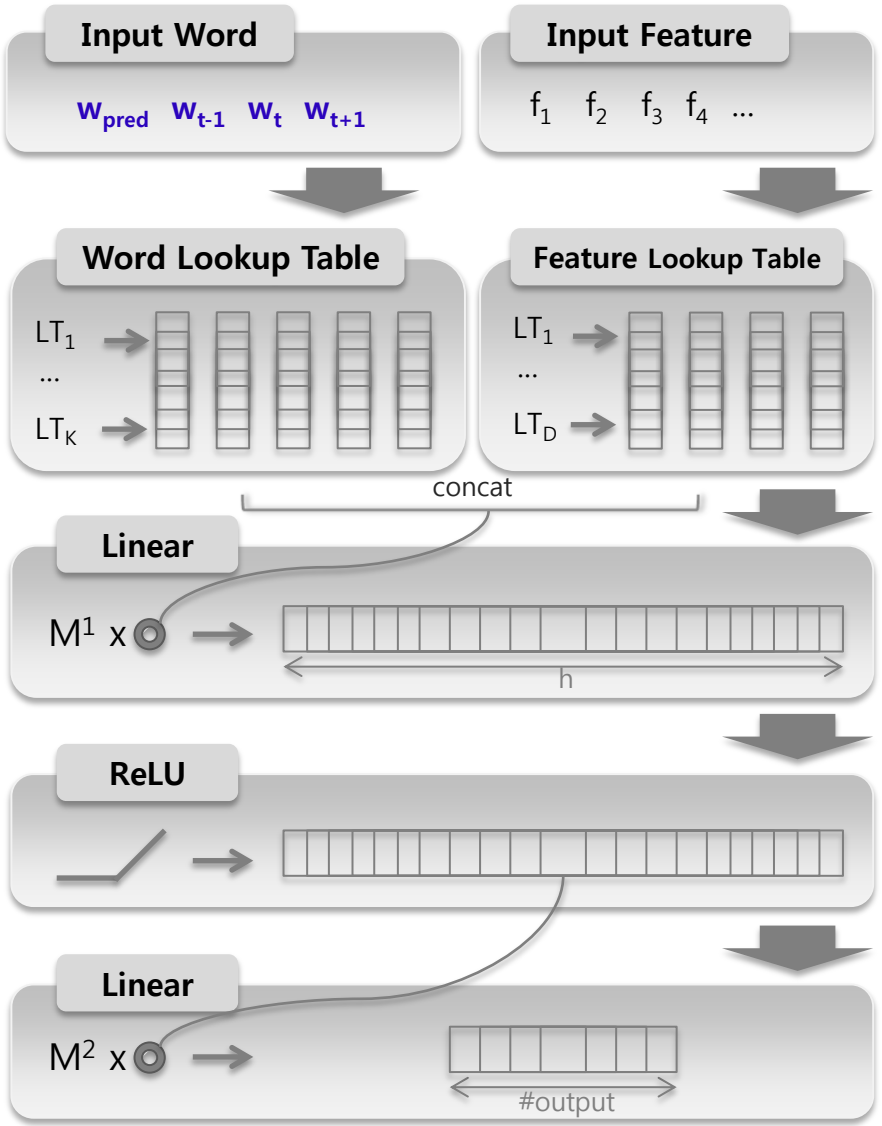


의존 구문 분석

의미역 결정

딥러닝 기반 한국어 의미역 결정 – KCC15(제출)

- Deep Learning
  - ReLU + Dropout
- Korean Word embedding
  - Predicate word, argument word
  - NNLM
- Feature embedding
  - POS, POS list
  - Dependency path, LCA
  - Distance, direction
- **성능 (AIC) – (개발 중) → RNN 적용 예정**
  - **F1: 75.14%**
    - Prec: 79.94%, Rec: 70.88%
  - **S-SVM 성능 (KCC14)**
    - **기본자질: F1 74.3%**
    - **기본자질+word cluster: 77.0%**
      - **정보과학회 논문지 2015.02**



한국어 상호참조해결

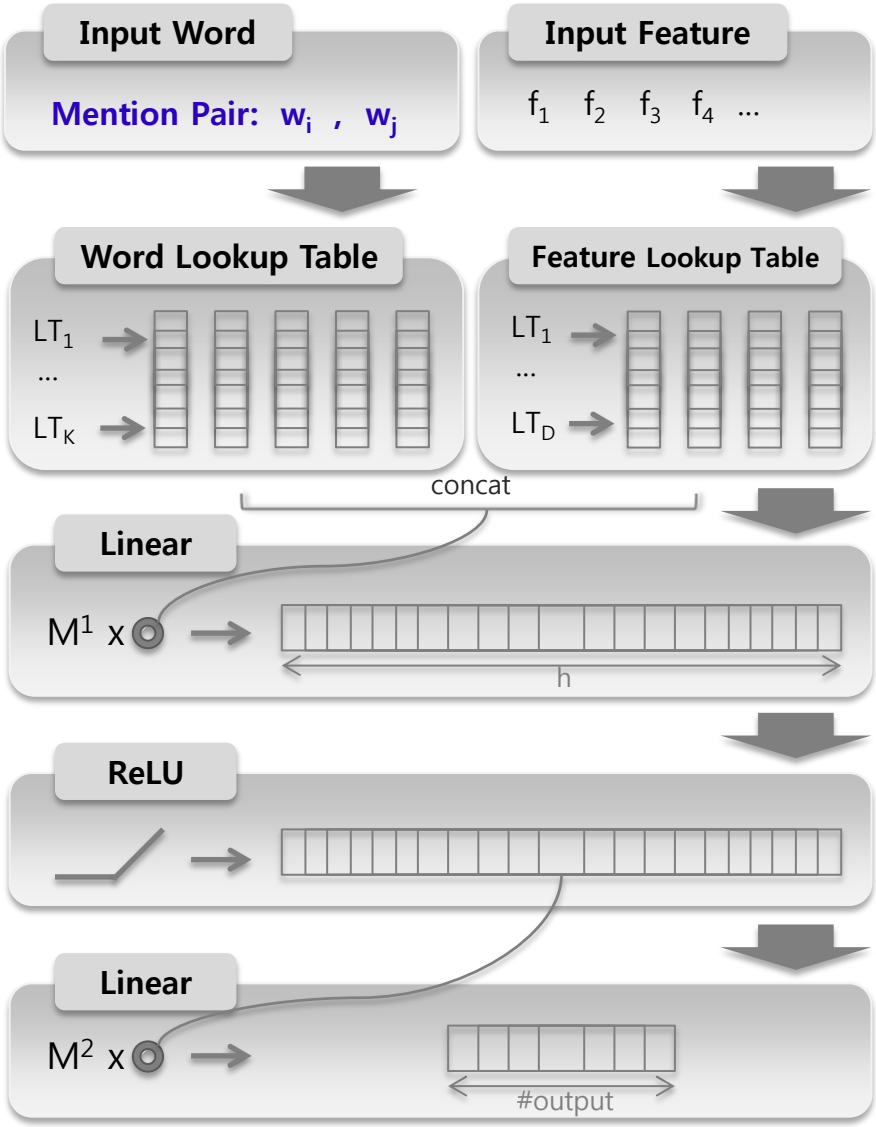
- **상호참조 (Coreference)**
  - 문서 내에서 이미 언급된 객체에 대하여 표현이 다른 단어로 다시 언급하는 것
  - Mention: 상호참조해결의 대상이 되는 모든 명사구(즉, 명사, 복합명사, 수식절을 포함한 명사구 등)를 의미
  - Entity: 상호참조가 해결된 Mention들의 집합
- **Mention Detection 예제**
  - **[[고양]에서 발생한 용오름]**은 **[토네이도]**와 같은 것으로 **[[[지상]의 뜨거운 공기]가 [[상층]의 찬 공기]와 갑자기 섞일 때]** 발생합니다.
  - **[뜨거운 공기]**가 빠르게 상승하고 **[찬 공기]**는 하강하면서 **[[길다란 기둥] 모양의 구름]**이 생겨나고 **[[그] 안]**에서 격렬한 **[회오리바람]**이 부는 겁니다.
- **Entity 예제**
  - **[지상의 뜨거운 공기], [뜨거운 공기]**
  - **[상층의 찬 공기], [찬 공기]**
  - **[길다란 기둥 모양의 구름], [그]**

상호참조해결 Mention-Pair 자질

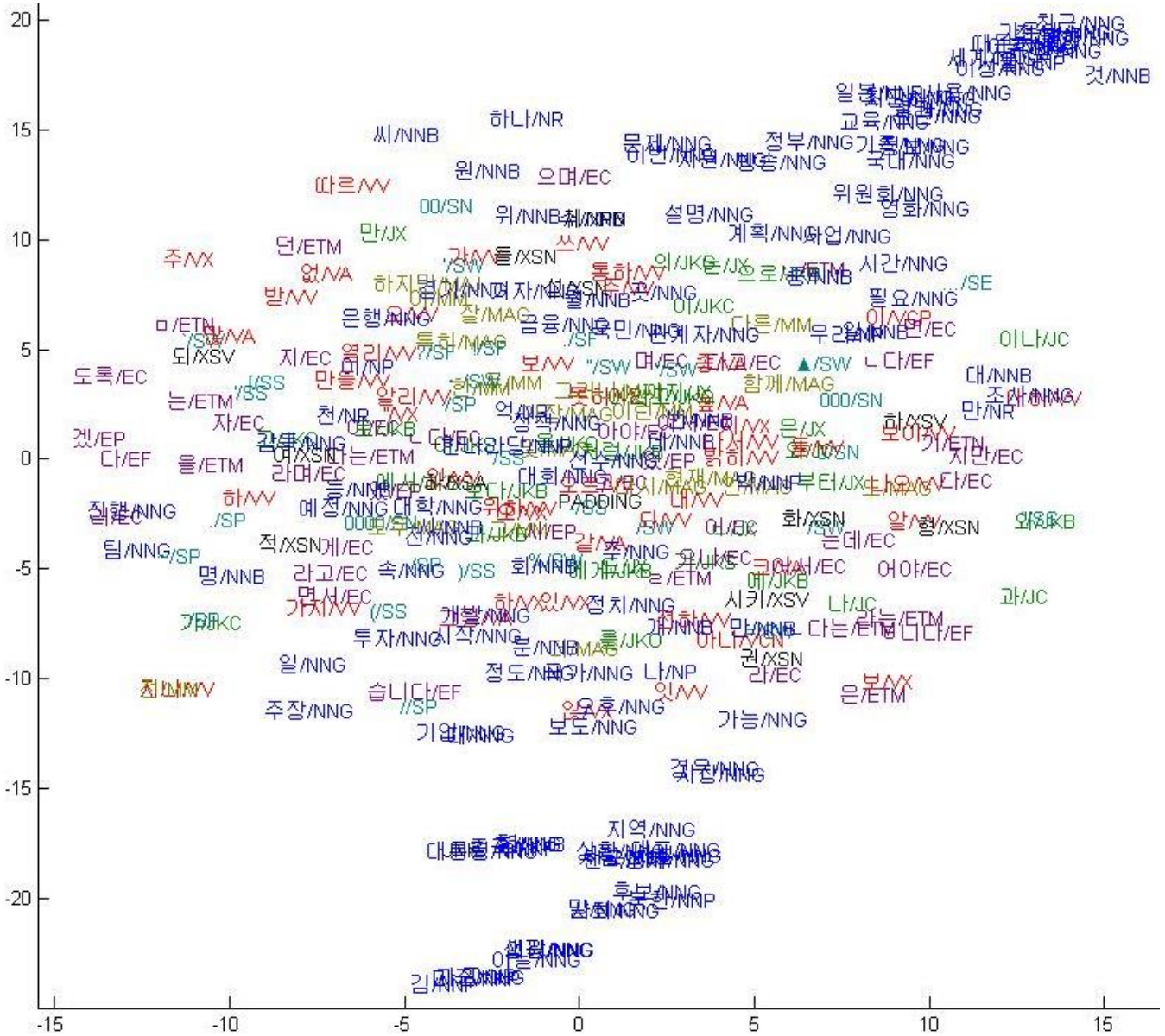
feature name	explanation	feature name	explanation
pronoun_1	Y if antecedent is a pronoun; else N	nonpro_str_match	C if both mentions are proper names and are the same string; else I
subject_1	Y if antecedent is a subject; else N	pn_str_match	C if the two mentions are both non-pronominal and are the same string; else I
left_pos_1	The POS tag of just before antecedent mention.	number	C if the mentions agree in number; I if they disagree
right_pos_1	The POS tag of just after antecedent mention.	animacy	C if the mentions match in animacy; else I
nested_1	Y if antecedent is a nested NP; else N	both_pronouns	C if both mentions are pronouns; I if neither are pronouns; else NA
number_2	SINGULAR or PLURAL, determined using a lexicon	both_proper_nouns	C if both mentions are proper nouns; I if neither are proper nouns; else NA
pronoun_2	Y if anaphoric is a pronoun; else N	span	word distance between the mentions
nested_2	Y if anaphoric is a nested NP; else N	semclass	if the mentions have the same semantic class; I if they don't
semclass_2	the semantic class of anaphoric; can be one of NA,human,thing,time,place by pronoun_dic and an NE recognizer.	distance	sentence distance between the mentions
animacy_2	Y if anaphoric is determined as HUMAN by pronoun_dic and an NE recognizer	same_mather	C if the mentions' dependency heads have same lemma; else I
left_pos_2	The POS tag of just before anaphoric mention.	number_12	SINGULAR or PLURAL of two mentions (e.g. number_12=singluarsingluar)
right_pos_2	The POS tag of just after anaphoric mention.	pronoun_12	the concatenation of the PRONOUN 2 feature values of two mentions.
head_match	C if the mentions have the same head noun; else I	nested_12	the concatenation of the NESTED 2 feature values of two mentions.
str_match	C if the mentions are the same string; else I	semclass_12	the concatenation of the SEMCLASS 2 feature values of two mentions.
subster_match	C if one mention is a substring of the other; else I	animacy_12	the concatenation of the ANIMACY 2 feature values of two mentions.
pro_str_match	C if both mentions are pronominal and are the same string; else I	guide	the result of Multi-Pass-Sieve

딥러닝 기반 한국어 상호참조해결 – KCC15(제출)

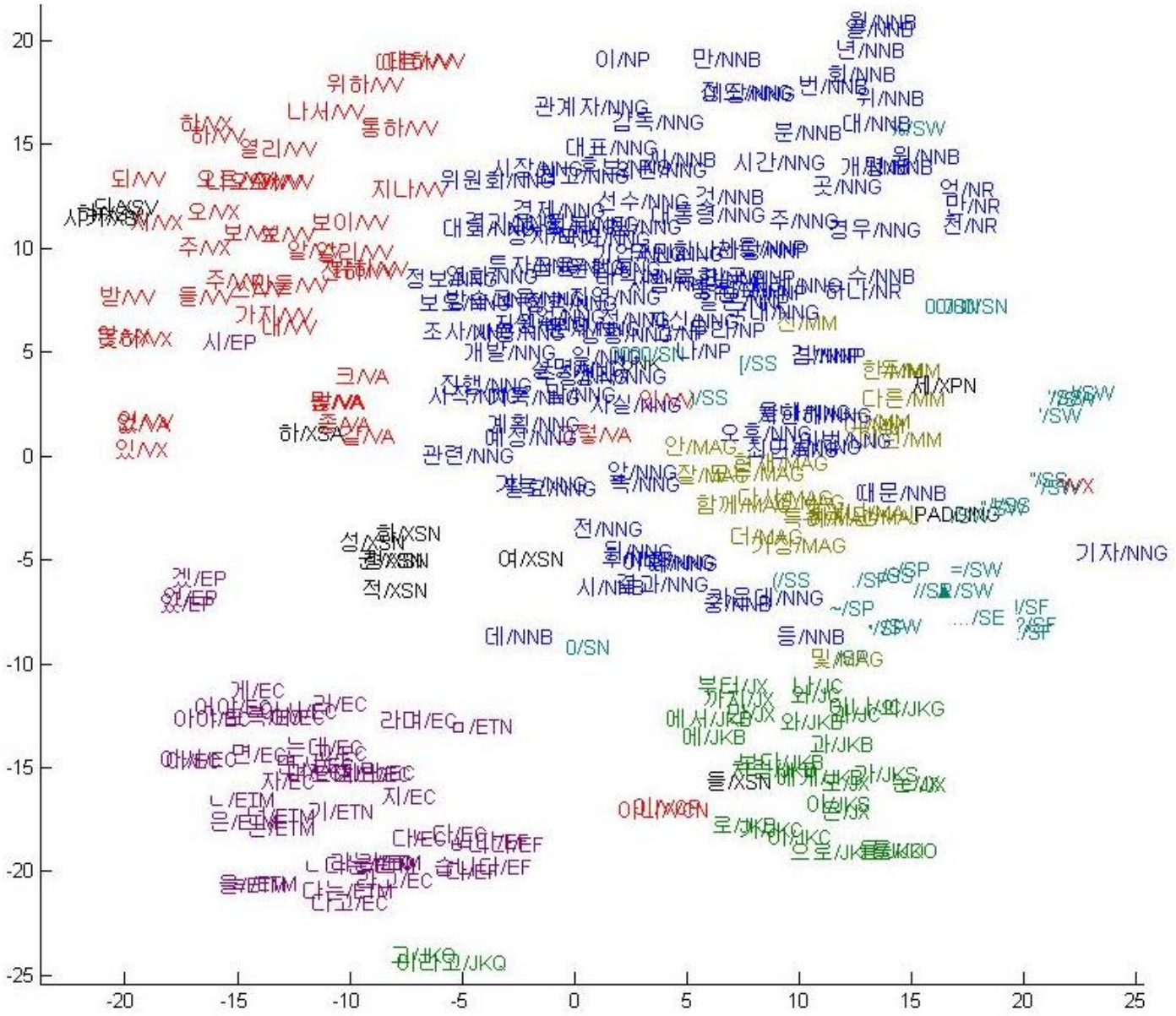
- **딥 러닝을 이용한 Mention-Pair model**
  - Korean word embedding + feature embedding (SVM과 동일)
  - Dropout + ReLU
- Data set
  - Test: 뉴스 1~20 문서
  - Training: 뉴스 21~100 문서, 장학퀴즈 QA 153개
- **성능**
  - **Deep Learning (MUC F1): 69.62%**
    - **No pre-training: F1 65.8%**
  - **SVM (MUC F1): 60.46%**



한국어 상호참조해결 – No Pre-training

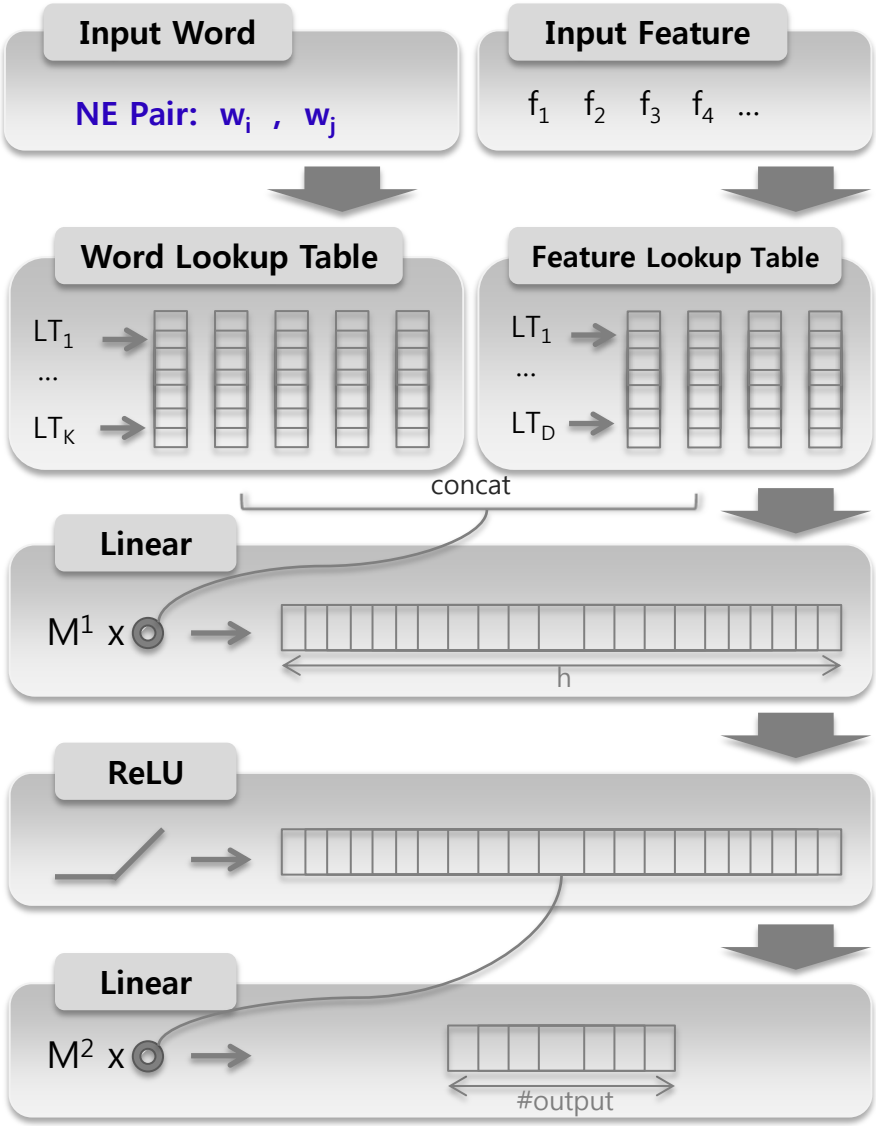


한국어 상호참조해결 – NNLM

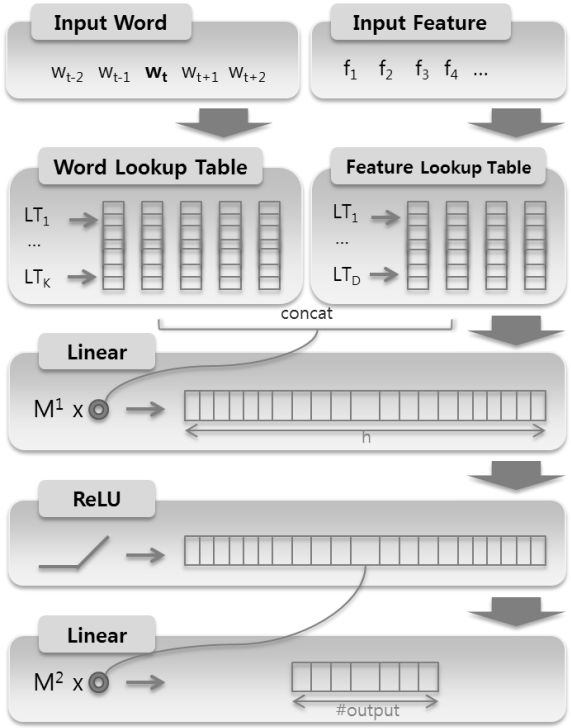


딥 러닝 기반 영어 관계 추출

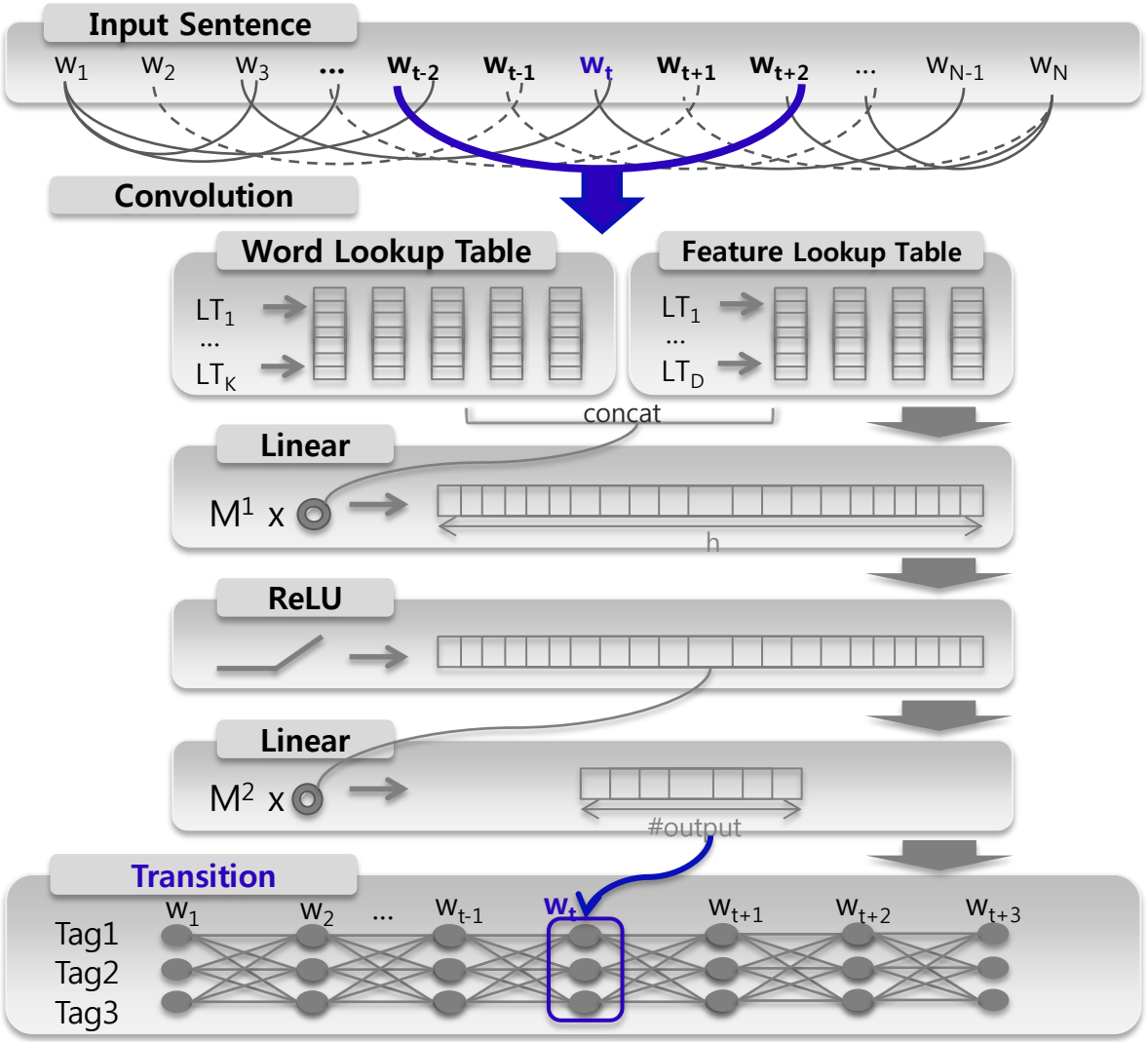
- 딥 러닝을 이용한 관계 추출
  - **Rel(e1, e2) → + or –**
  - Word embedding: word pair, words around word pair
  - Feature embedding
    - POS, NE, dependency path
  - Dropout + ReLU
- Data set
  - Conll04.corp: 5 relation types
- **성능 (개발 중) → CNN과 결합된 모델 적용 예정 중**
  - **F1=66.86%**
    - **Prec=83.03, Rec=55.97**
  - **S-SVM (F1): 72.25%**



영어/한국어 개체명 인식 – FFNN, CNN (정보과학회 동계학술대회14)



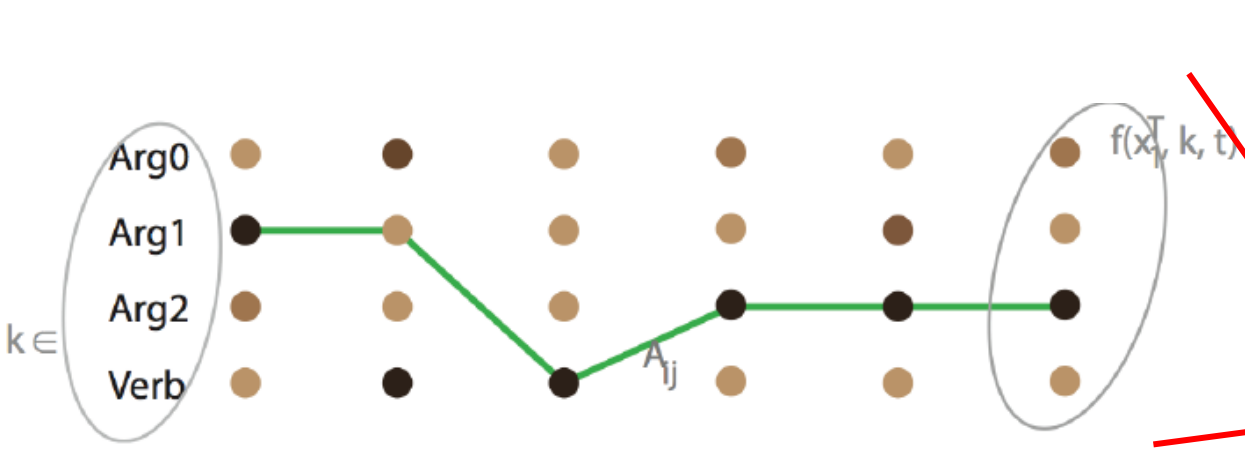
Feed-Forward NN

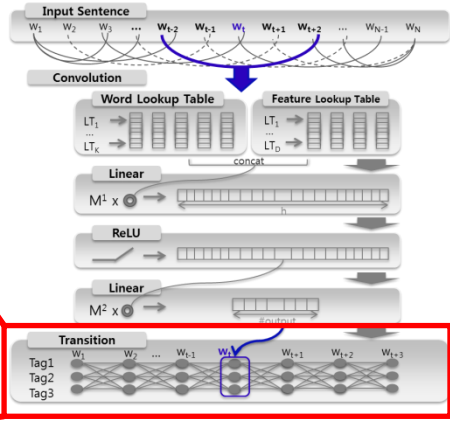


Convolutional NN

Transition Layer

- Word(t-th)-tag(k) score: $f([\boldsymbol{x}]_1^T, k, t, \boldsymbol{\theta})$
- Transition score A_{kl}: jump from tag k to l





- **Sentence score**:

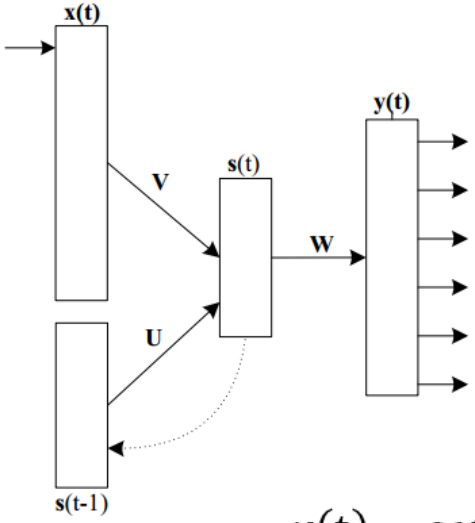
$$s([\boldsymbol{x}]_1^T, [i]_1^T, \tilde{\boldsymbol{\theta}}) = \sum_{t=1}^{T} \left(A_{[i]_{t-1}[i]_t} + f([\boldsymbol{x}]_1^T, [i]_t, t, \boldsymbol{\theta}) \right)$$

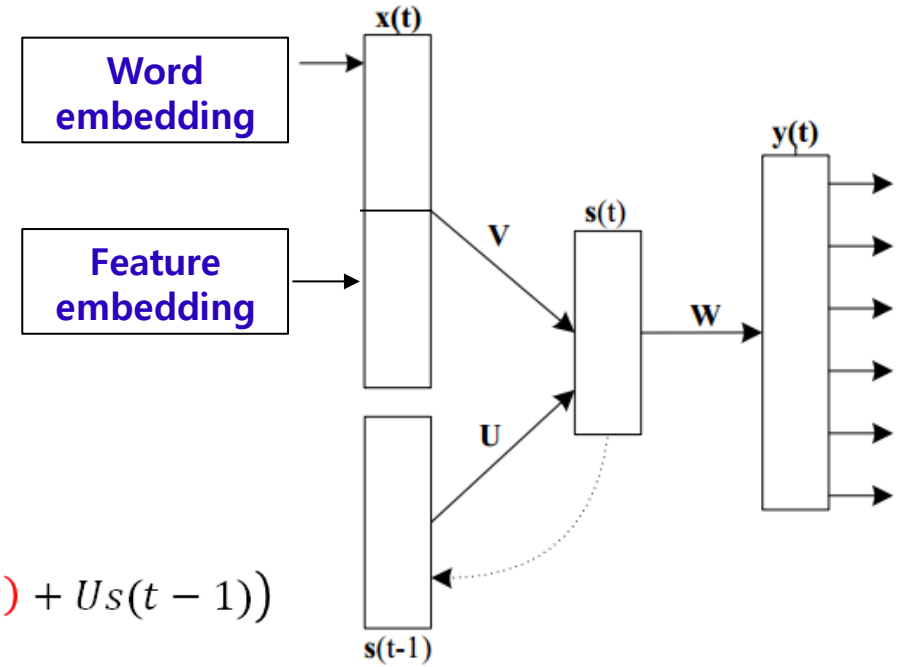
개체명 인식 실험

영어 개체명 인식 (BBN data set)	F1
S-SVM1 (base + NE dic. feature)	90.66
S-SVM2 (base + NE dic. + word cluster feature)	90.87
S-SVM3 (base + NE dic. + word cluster + word embedding feature)	91.74
NN1 (Sigmoid + Pre-training)	88.19
NN2 (ReLU + Pre-training)	88.31
NN3 (ReLU + Dropout + Pre-training)	90.01
CNN4SEQ (ReLU + Dropout + Pre-training)	91.40

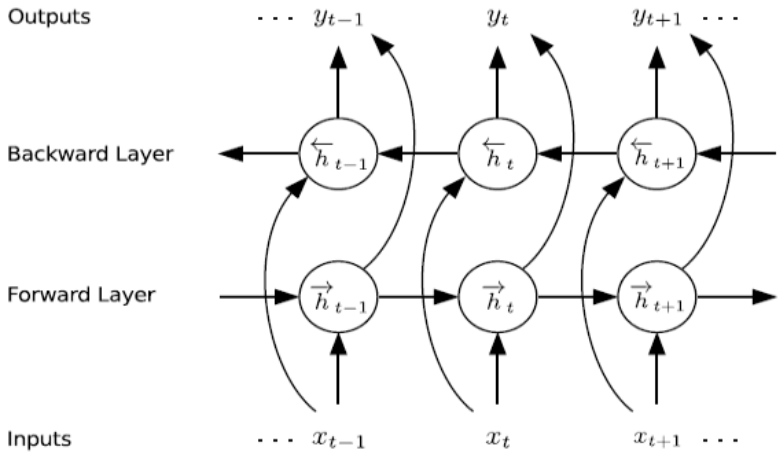
한국어 개체명 인식 (TV 도메인)	F1
S-SVM1 (base + NE dic. feature)	87.77
S-SVM2 (base + NE dic. + word cluster feature)	88.40
S-SVM3 (base + NE dic. + word cluster feature + morpheme feature)	89.03
NN (ReLU + Dropout + Pre-training)	87.70
CNN4SEQ (ReLU + Dropout + Pre-training)	88.43

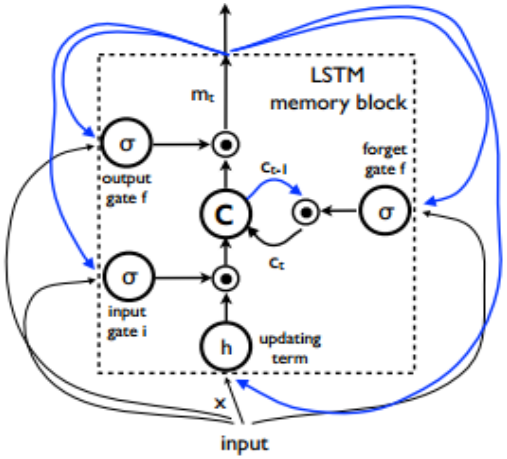
개체명 인식, NLU – RNN, LSTM





- $y(t) = sotfmax(Ws(t))$
- $s(t) = sigm\left(V_x x(t) + V_f f(x) + Us(t-1)\right)$





개체명 인식, NLU 실험

NLU (Airplane Traveling Information System data)	F1
NN (Sigm + Dropout + Word embedding)	91.73
RNN (Sigm + Dropout + Word embedding)	94.62
Bidirectional RNN (Sigm + Dropout + Word embedding)	94.73
LSTM RNN (Sigm + Dropout + Word embedding)	94.75
GRU RNN (Sigm + Dropout + Word embedding)	94.89
CNN4SEQ (Sigm + Dropout + Word embedding)	95.01

영어 개체명 인식 (CoNLL03 data set)	F1
Structural SVM (baseline feature)	82.77
Structural SVM (baseline feature + Word embedding feature)	85.58
NN (Sigm + Dropout + Word embedding)	86.23
RNN (Sigm + Dropout + Word embedding)	86.92
Bidirectional RNN (Sigm + Dropout + Word embedding)	87.82
LSTM RNN (Sigm + Dropout + Word embedding)	?
GRU RNN (Sigm + Dropout + Word embedding)	87.09
CNN4SEQ (Sigm + Dropout + Word embedding)	89.09

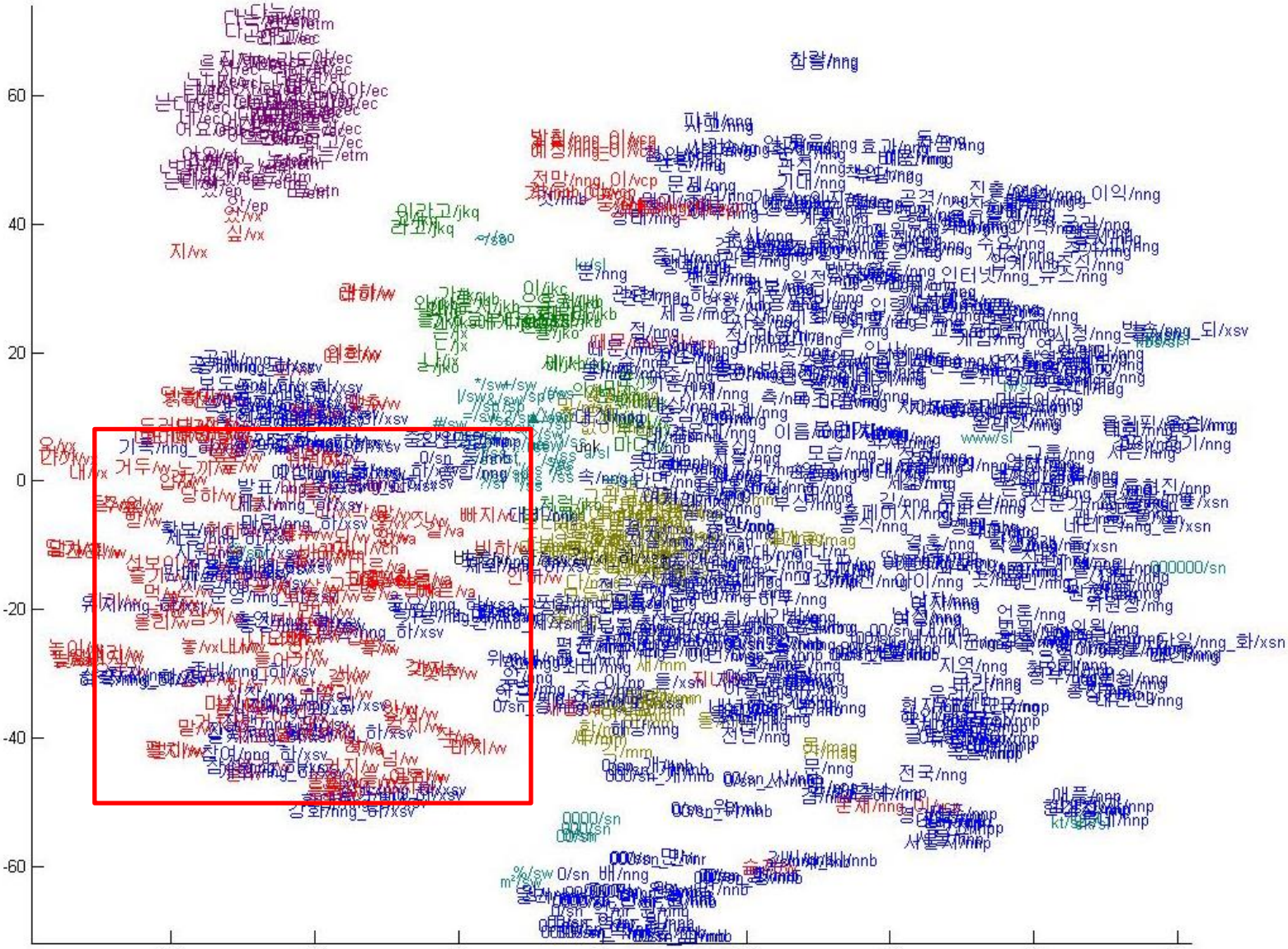
차례

- 딥러닝 소개
- Word Embedding
- Word Embedding 응용
  - 딥러닝 기반의 자연어처리
- **Phrase/Sentence Embedding**

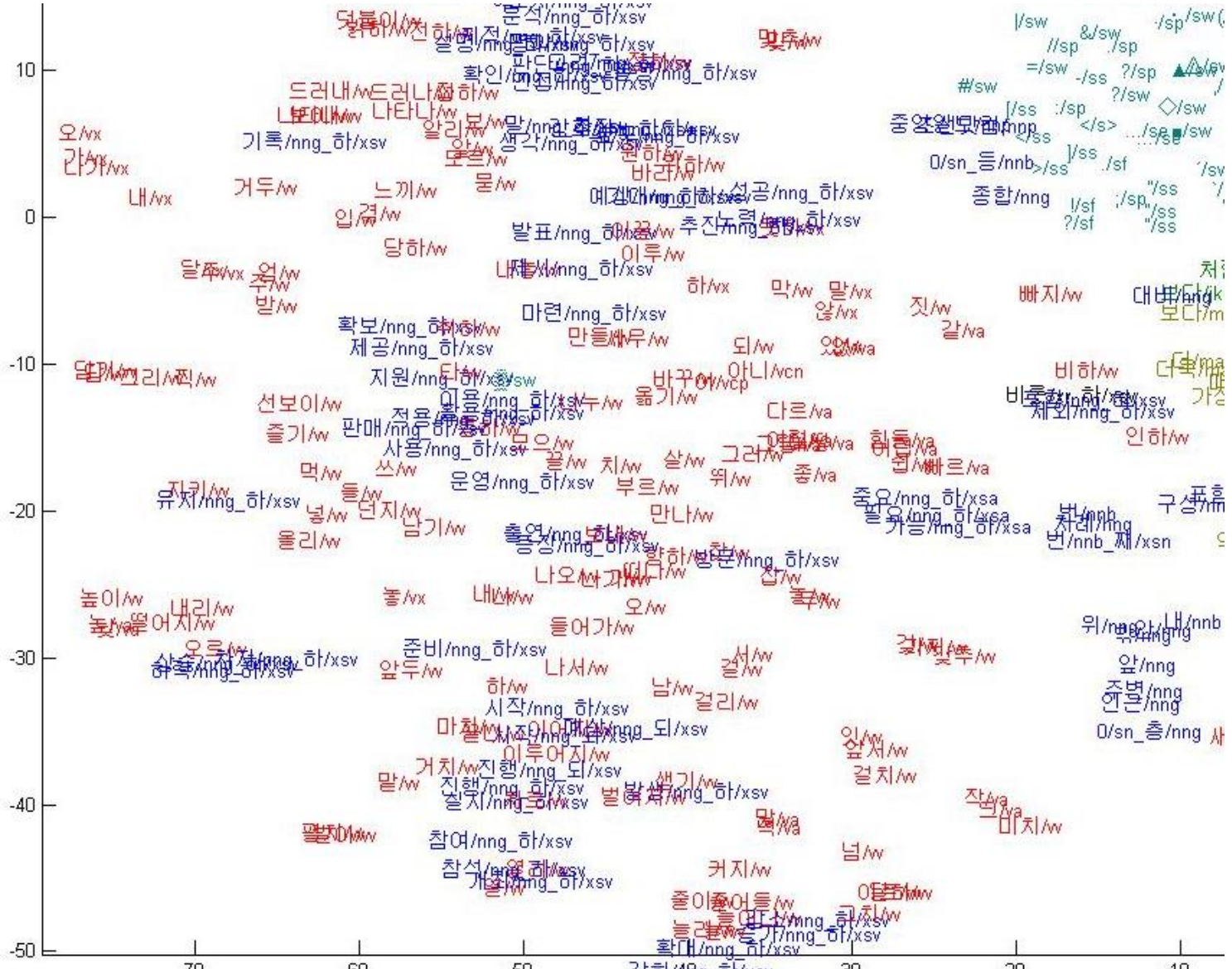
한국어 Phrase Embedding – Word2Vec

- **입력: 형태소 → 형태소(_형태소)***
  - 어절에서 형태소 패턴 이용 (MI 등을 이용할 수도 있음)
- '**서울/nnp**'와 비슷한 word/phrase
  - 강남구/nnp 0.825 (cosine)
  - **목동/nnp_sbs/sl** 0.804
  - 소동동/nnp 0.793
  - 대치쌍용/nnp 0.791
  - **마천/nnp_시장/nng** 0.789
- '**줄/vv**'와 비슷한 word/phrase
  - 늘/vv 0.936 (cosine)
  - 줄어들/vv 0.934
  - **감소/nng_하/xsv** 0.918
  - **급감/nng_하/xsv** 0.900
  - **증가/nng_하/xsv** 0.873

한국어 Phrase Embedding – NNLM

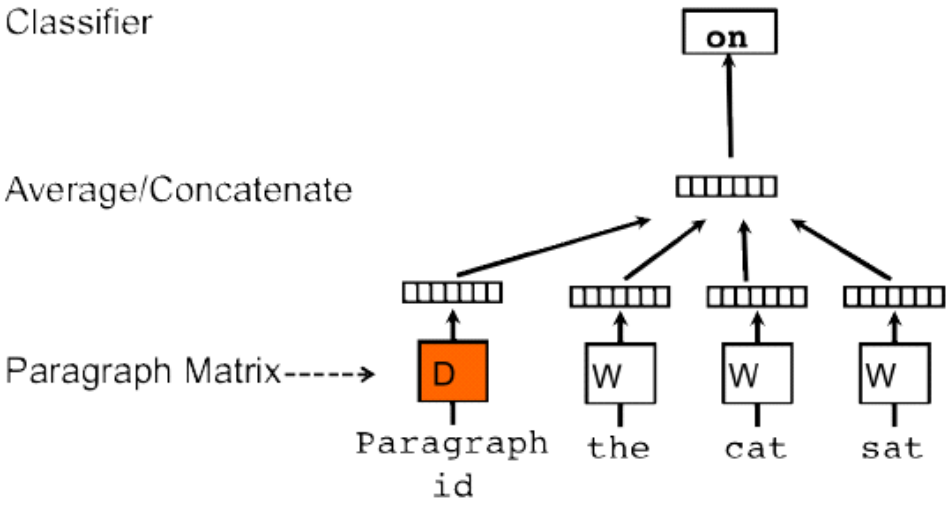


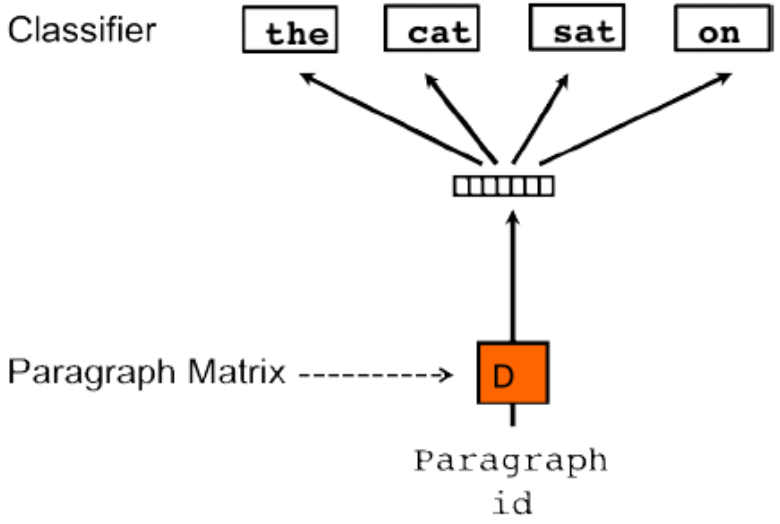
한국어 Phrase Embedding – NNLM (VV)



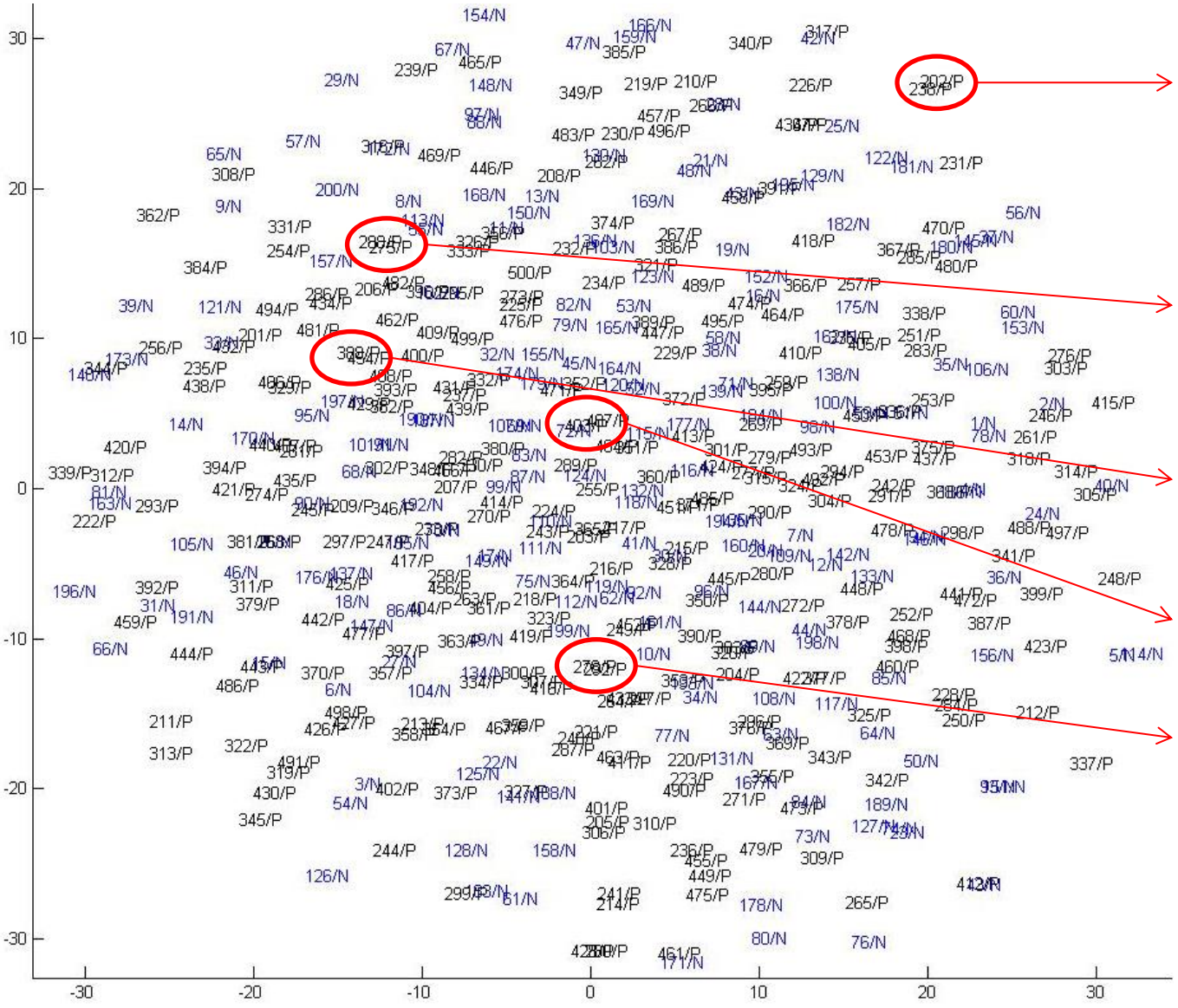
Paragraph Vector (ICML14)

- Distributed Representations of Sentences and Documents





한국어 Paragraph Vector 실험



202: 옵니아보다 빠른 촬영속도 캠코더 기능으로 동영상 촬영 만족스러움
238: 카메라화질도 깔끔하고 기능도 다 잘갖춘것같애요

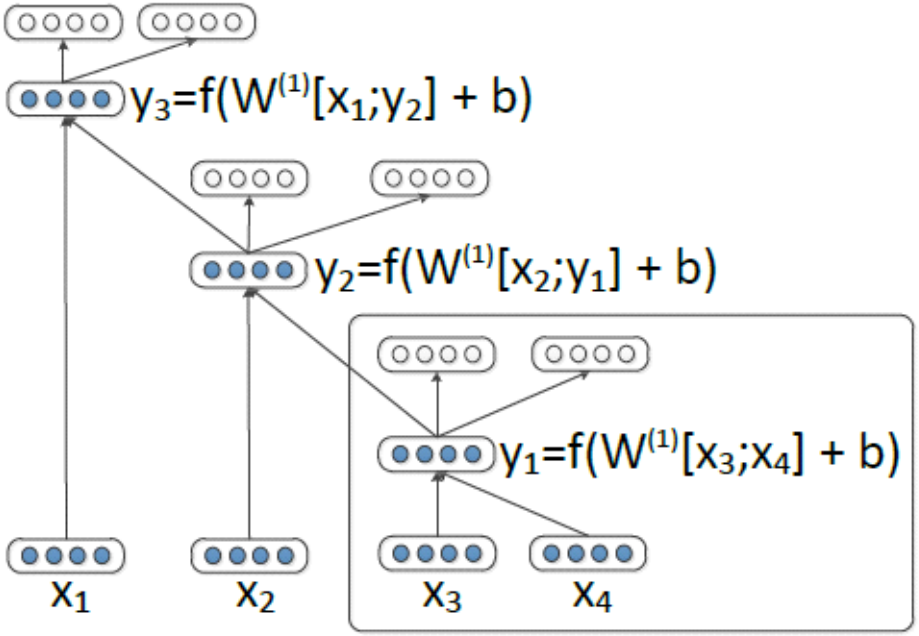
275: 정말 좋은 제품을 싸게 구입했어요
288: 빠른속도도 마음에들고

388: 디자인 좋고 기능 어플로 다해결 됩니다
454: 디자인 완전 대박입니다

403: 깔끔하네요
487: 너무 귀엽고 쉽고 좋아요

278: 반응속도가 정말 빠르군요
292: 확실히 2.2가 빠릅니다

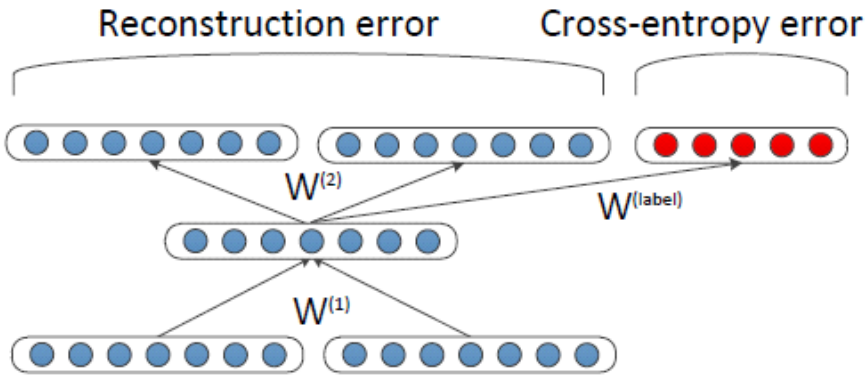
Recursive Autoencoder+Sentiment Analysis (EMNLP11)



$$p = f(W^{(1)}[c_1; c_2] + b^{(1)}),$$

$$[c_1'; c_2'] = W^{(2)}p + b^{(2)}.$$

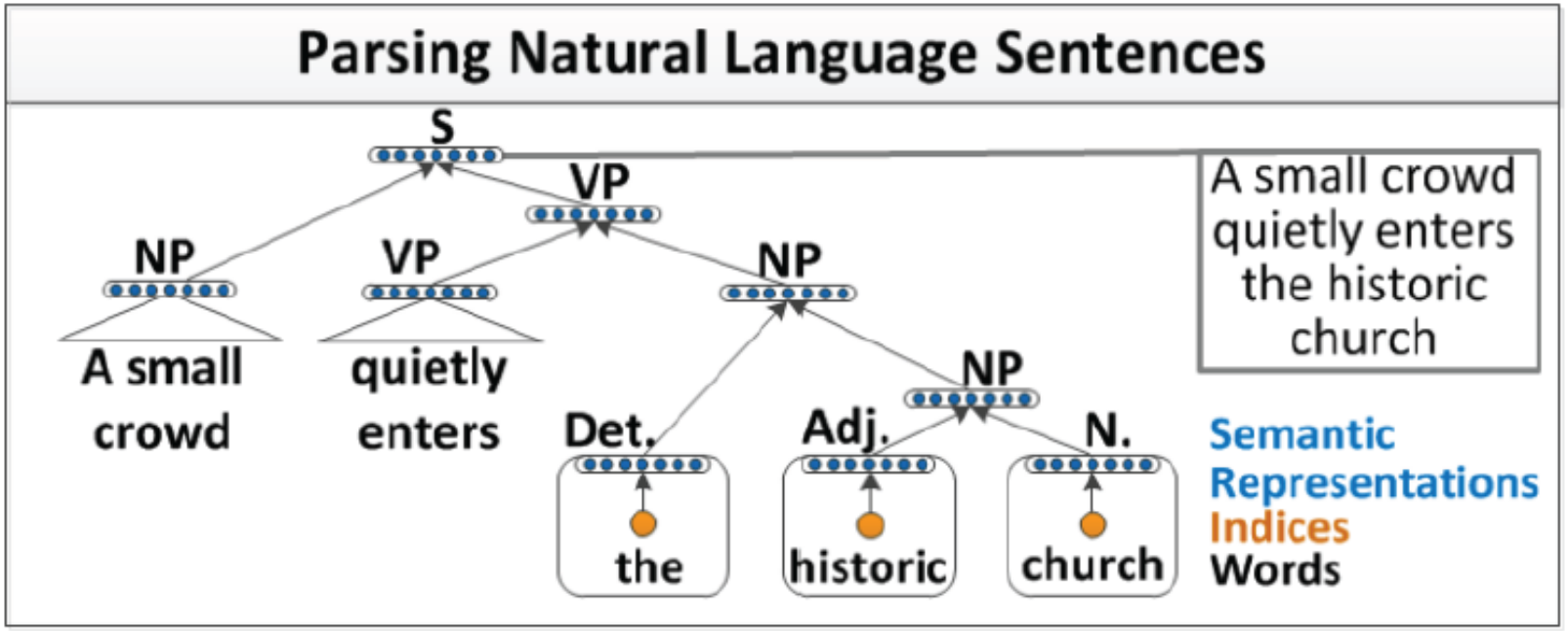
$$E_{rec}([c_1; c_2]) = \frac{1}{2}\left|\left|[c_1; c_2] - [c_1'; c_2']\right|\right|^2.$$

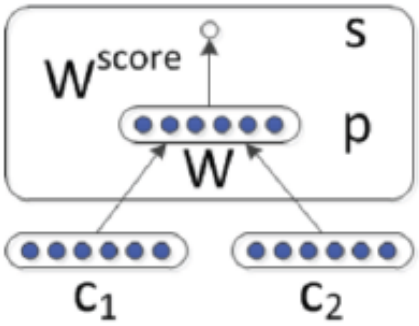


$$E([c_1; c_2]_s, p_s, t, \theta) =$$

$$\alpha E_{rec}([c_1; c_2]_s; \theta) + (1-\alpha)E_{cE}(p_s, t; \theta).$$

Recursive Neural Network (ICML13)

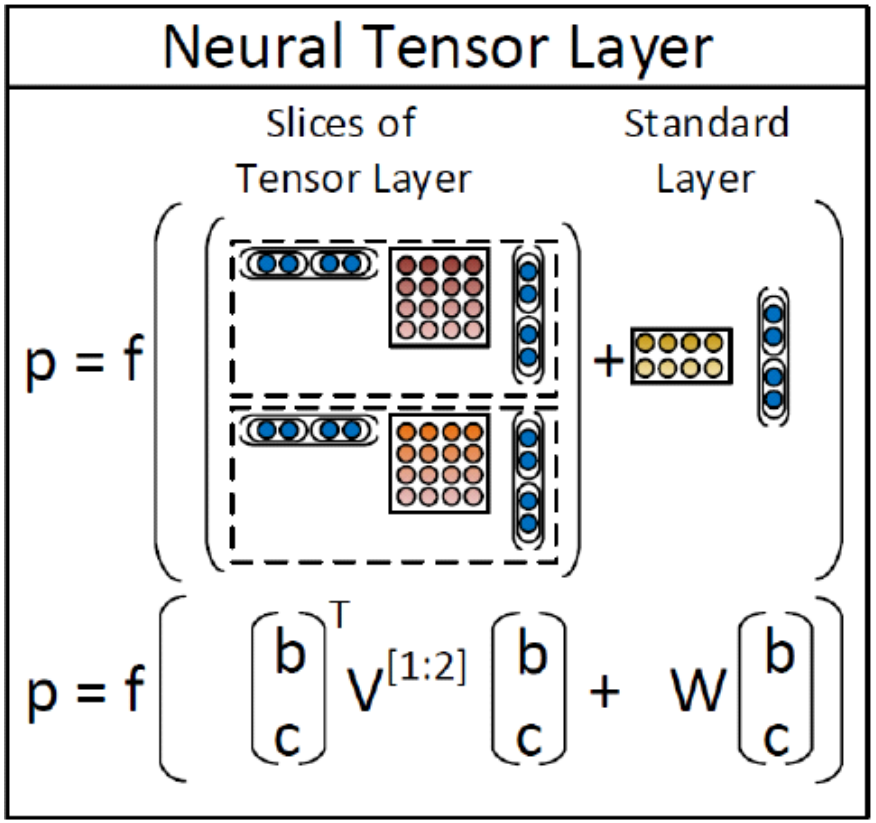


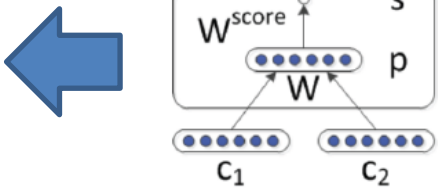


$$s = W^{score} p \qquad (9)$$

$$p = f(W[c_1; c_2] + b)$$

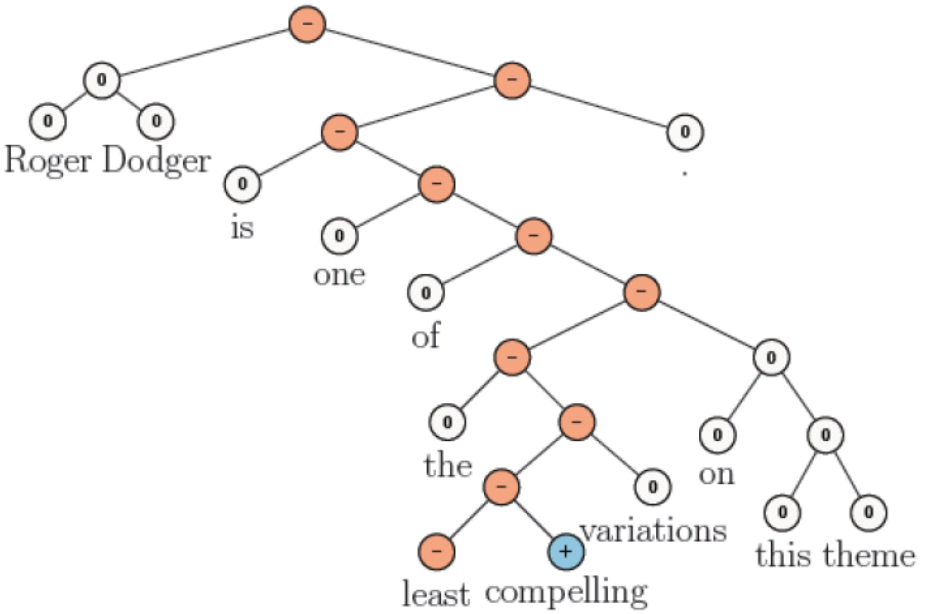
Recursive Neural Tensor Networks (EMNLP13)





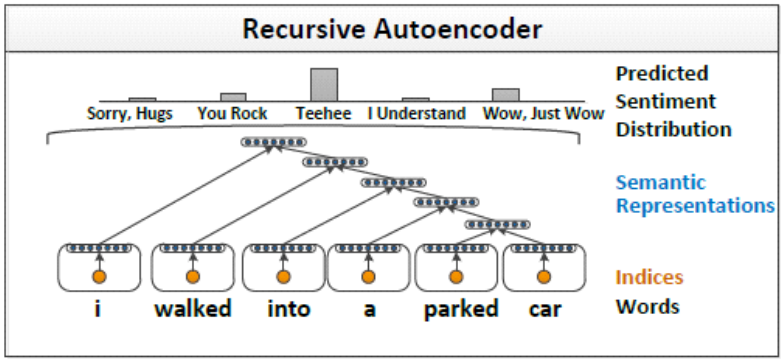
$$s = W^{score} p \quad (9)$$

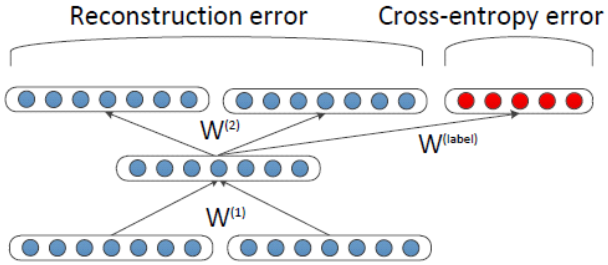
$$p = f(W[c_1; c_2] + b)$$



한국어 감성 분석 – Recursive Autoencoder

- Mobile data
  - Train: 4543, Test: 500
- Recursive Autoencoder 적용
  - **구문분석 결과 적용 예정**
  - Word embedding
    - 한국어 10만 단어 + 도메인 특화 1420 단어

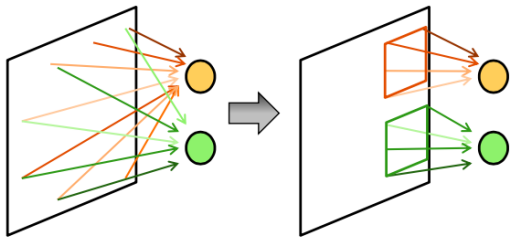


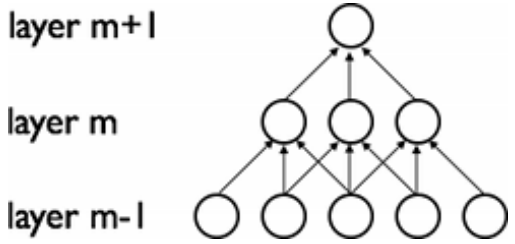


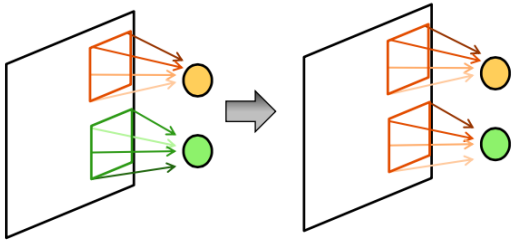
Data set	Model	Accuracy
Mobile Train: 4543 Test: 500	**SVM (word feature)**	**85.58**
	RAE (word feature) + Word embedding	**87.57**

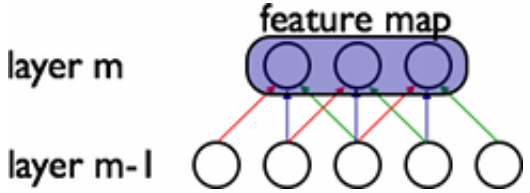
CNN for Sentence Classification (Sentiment Analysis)

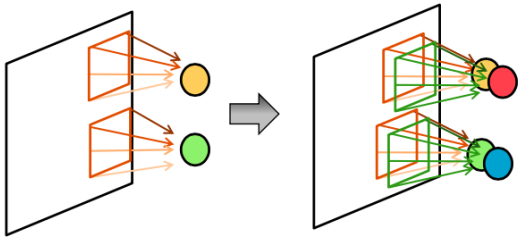
- Convolutional NN
  - Convolution Layer
    - Sparse Connectivity
    - Shared Weights
    - Multiple feature maps
  - Sub-sampling Layer
    - Average/max pooling
    - NxN→1
- NLP (Sentence Classification)에 적용
  - ACL14
  - EMNLP14



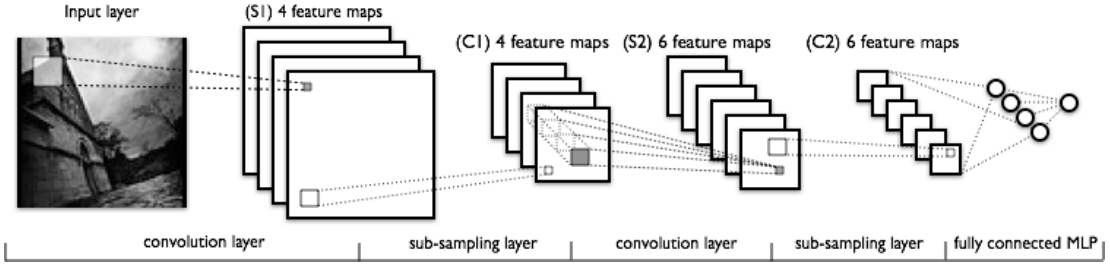




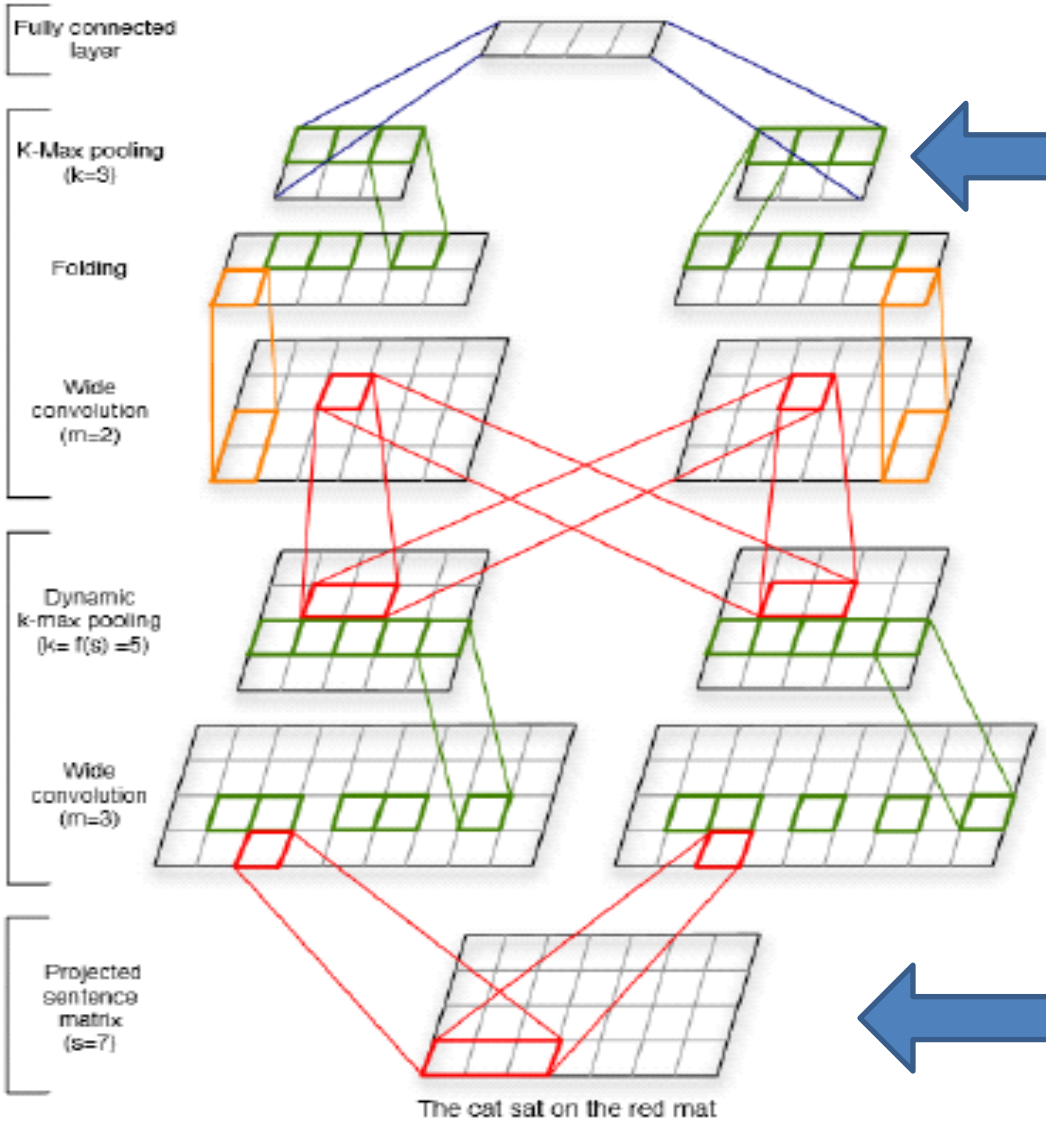




Multiple feature maps



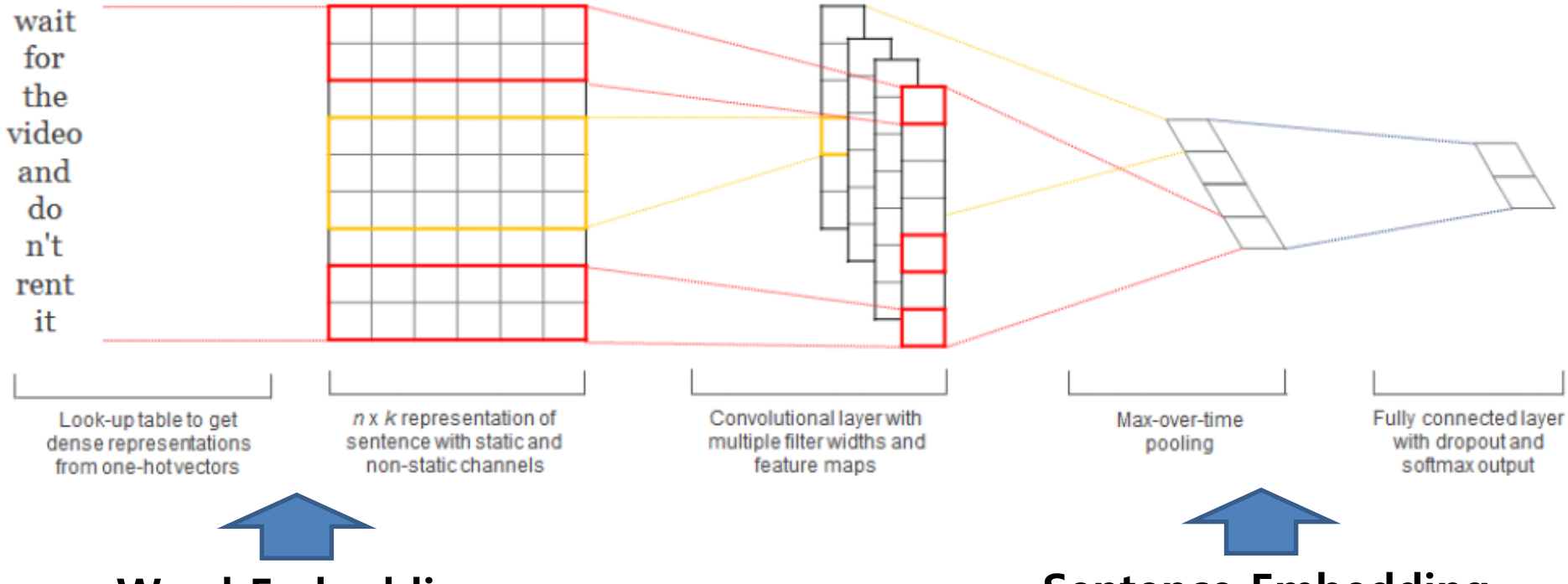
Convolutional Neural Network for Modeling Sentences (ACL14)



Sentence Embedding

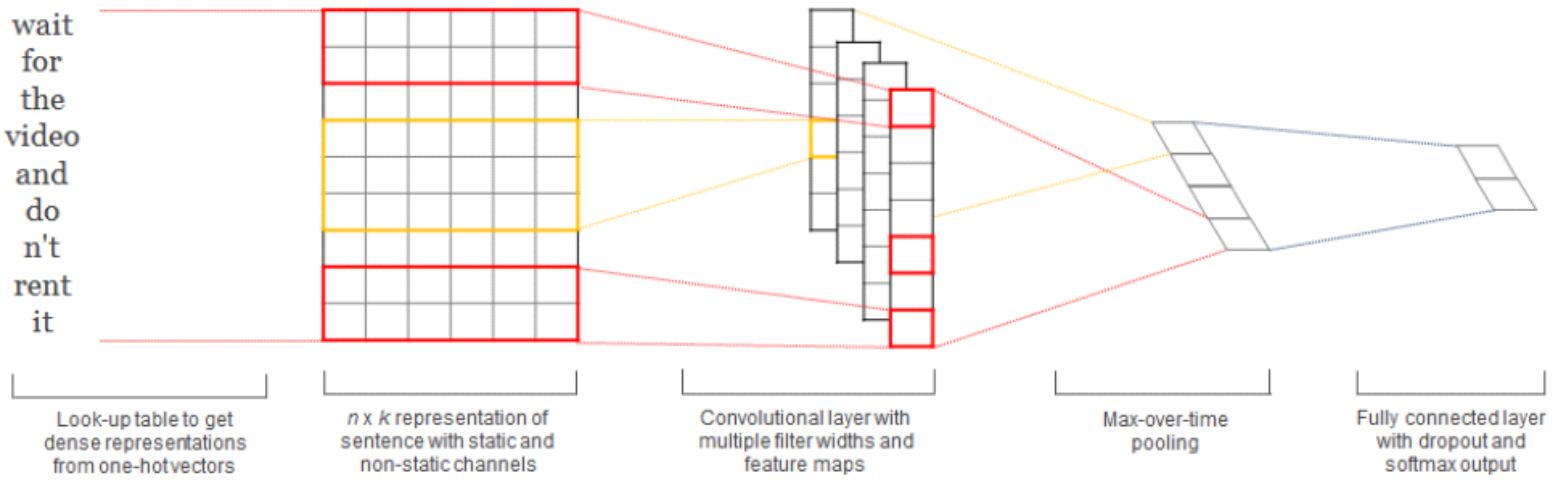
Word Embedding

Convolutional Neural Network for Sentence Classification (EMNLP14)



한국어 감성 분석 – CNN

- Mobile data
  - Train: 4543, Test: 500
- EMNLP14 모델(CNN) 적용
  - Matlab으로 구현
  - Word embedding
    - 한국어 10만 단어 + 도메인 특화 1420 단어



한국어 감성 분석 – CNN : 실험

Data set	Model	Accuracy
Mobile Train: 4543 Test: 500	**SVM (word feature)**	**85.58**
	RAE (word feature))+Word embedding	**87.57**
	CNN(relu,kernel3,hid50)+Word embedding (word feature)	**91.20**
	CNN(relu,kernel3,hid50)+Random init.	**89.00**