Deep Learning and Natural Language Processing

Natural Language Processing & AI Lab.
Dept. of Computer Science & Eng.
limhseok@korea.ac.kr
Heuiseok Lim

KOREA UNIVERSITY

Natural Language Processing & Artificial Intelligence

NLP Basics

Overview

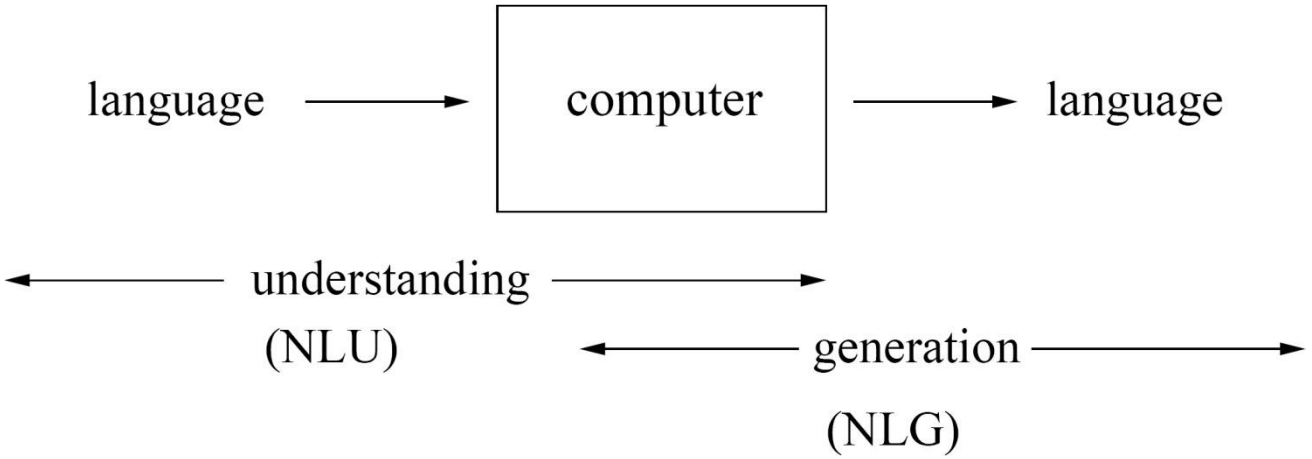
What is NLP?(1/2)

- Definition of Natural Language Processing (NLP)
  - This is a difficult question to answer because "*there are almost as many definit ions as the number of researchers studying it*" (Obermeier, 1988)

    - *The branch of information science that* ***deals with natural language informa tion*** (자연어 정보를 취급하는 정보과학의 한 분야)
    - *The formulation and investigation of* ***computationally effective mechanisms f or communication through natural language*** (자연어를 통한 의사소통의 계 산학적으로 효과적인 수단)
    - *A subfield of artificial intelligence and linguistics for* ***making computers "understand" statements written in human languages*** (인간의 언어로 쓰여진 문서를 이해하는 컴퓨터)
- Two motivations for NLP (Allen, 1994)
  - The ***scientific or linguistic motivation*** is to understand the nature of language through the tools provided by computer science
  - The ***technological motivation*** is to improve communication between humans and machines

What is NLP?(2/2)

- One simple (but practical) answer
  - Computer using natural language as input and/or output



  - Or components enabling such a computer

Overview of Basic NLP

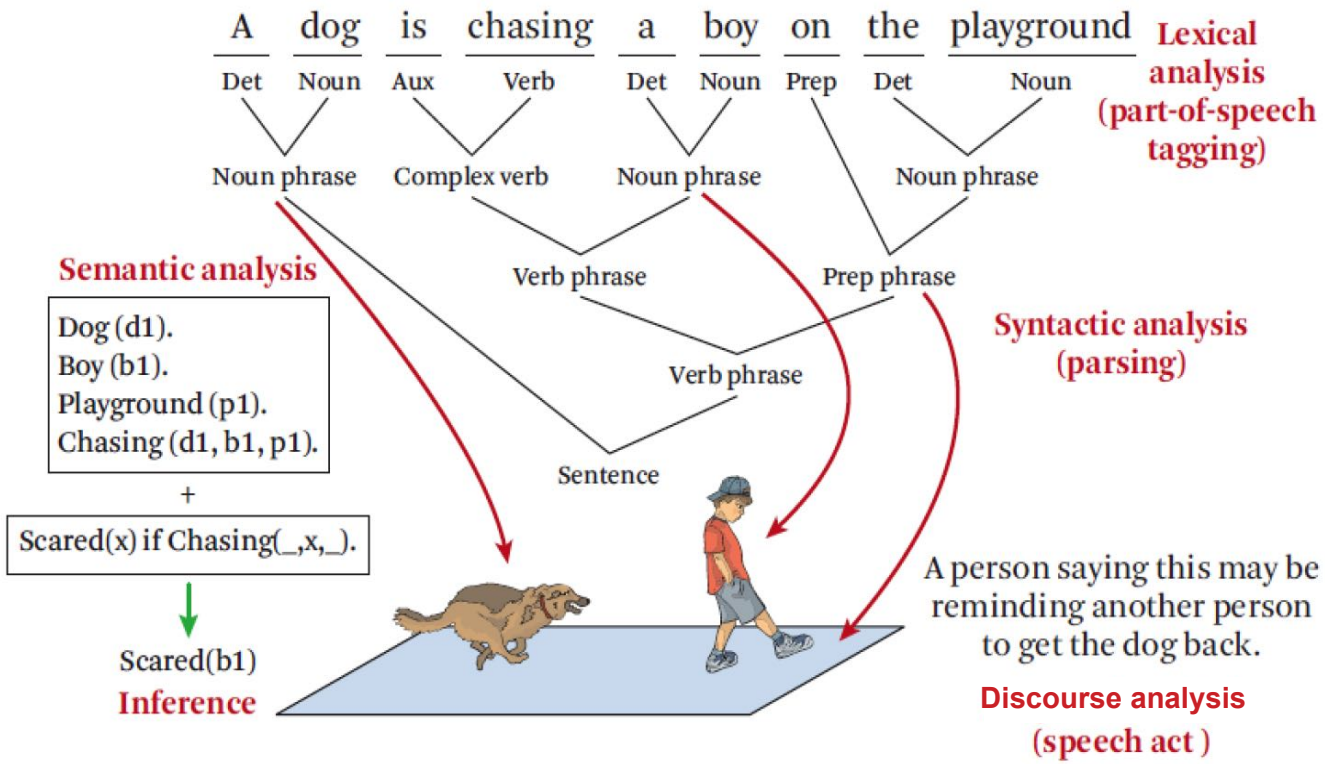
General steps in NLP

- Preprocessing
- Lexical Analysis
  - Word(lexicon), Morphology, word segmentation
- Syntactic Analysis
  - Sentence structure, phrase, grammar,…
- Semantic Analysis
  - Meaning, execute commands
- Discourse Analysis
  - Meaning of a text
  - Relationship between sentences

    Ex) I disagree and so does John. (does-> disagree)

Overview of Basic NLP

- General steps in NLP



History of NLP

- Research on NLP began about 50 years ago
  - Also many people have imagined a conversational system in the future
  - One famous script in the old movie
    - Dave Bowman:
      - “Open the pod bay doors, HAL”
    - HAL:
      - “I’m sorry Dave,
      - I’m afraid I can’t do that”
  - But still, we can not build such an intelligent dialog based system

Why NLP is hard?

- A NLP system needs to answer the question "who did what to whom"
- **Language is ambiguous**
  - At all levels: lexical, phrase, semantic
  - Iraqi Head Seeks Arms
    - Word sense is ambiguous (head, arms)
  - Stolen Painting Found by Tree
    - Thematic role is ambiguous: tree is agent or location?
  - Ban on Nude Dancing on Governor's Desk
    - Syntactic structure (attachment) is ambiguous: is the ban or the dancing on the desk?
  - Hospitals Are Sued by 7 Foot Doctors
    - Semantics is ambiguous : what is 7 foot?

Why NLP is hard?

- Language is flexible
  - New words, new meanings
  - Different meanings in different contexts
- Language is subtle
  - He arrived at the lecture
  - He chuckled at the lecture
  - He chuckled his way through the lecture
  - **He arrived his way through the lecture
- Language is complex!

Why NLP is hard?

- MANY hidden variables
  - Knowledge about the world
  - Knowledge about the context
  - Knowledge about human communication techniques
    - Can you tell me the time?
- Problem of scale
  - Many (infinite?) possible words, meanings, context
- Problem of sparse data
  - Very difficult to do statistical analysis, most things (words, concepts) are never seen before
- Long distance correlations

Why NLP is hard?

- Key problems:
  - Ambiguity of Language

[One example from L.Lee]

"At last, a computer that understands you like your mother"

Ambiguity

"At last, a computer that understands you like your mother"

- Possible interpretations
  - A computer understands you as your mother understands you
  - (어머니가 당신을 이해하는 것처럼 당신을 이해하는 컴퓨터)
  - It understands (that) you like your mother
  - (당신이 당신의 어머니를 좋아한다는 사실을 컴퓨터가 이해한다.)
  - It understands you as it understands your mother
  - (컴퓨터가 당신의 어머니를 이해하는 것처럼 당신도 이해한다.)

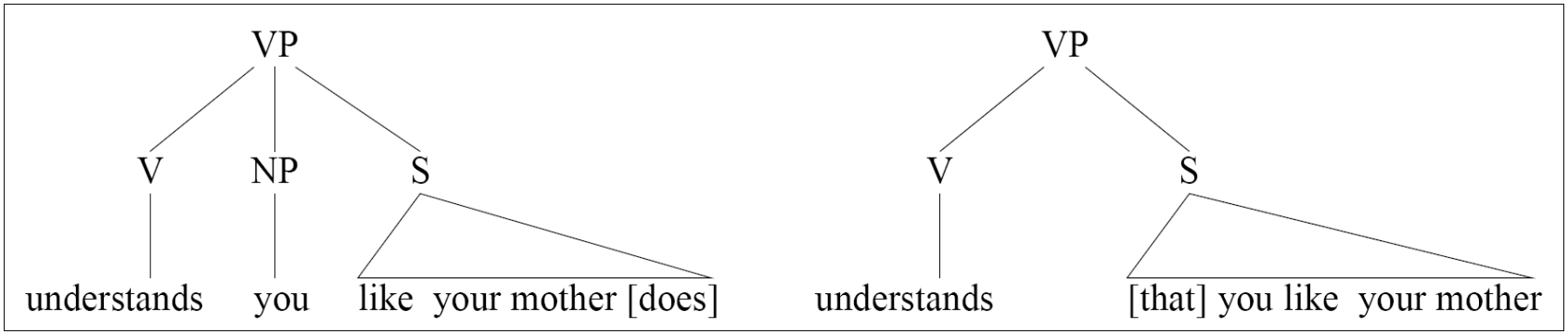
  - What is the correct interpretation?

Ambiguities at Multi Levels (1/3)

- At the word level (word ambiguity)
  - “At last, computer understand you *like* your mother”
  - Part of speech tag of the word “Like”
    - can be [Verb]
    - Or can be [Preposition]

Ambiguities at Multi Levels (2/3)

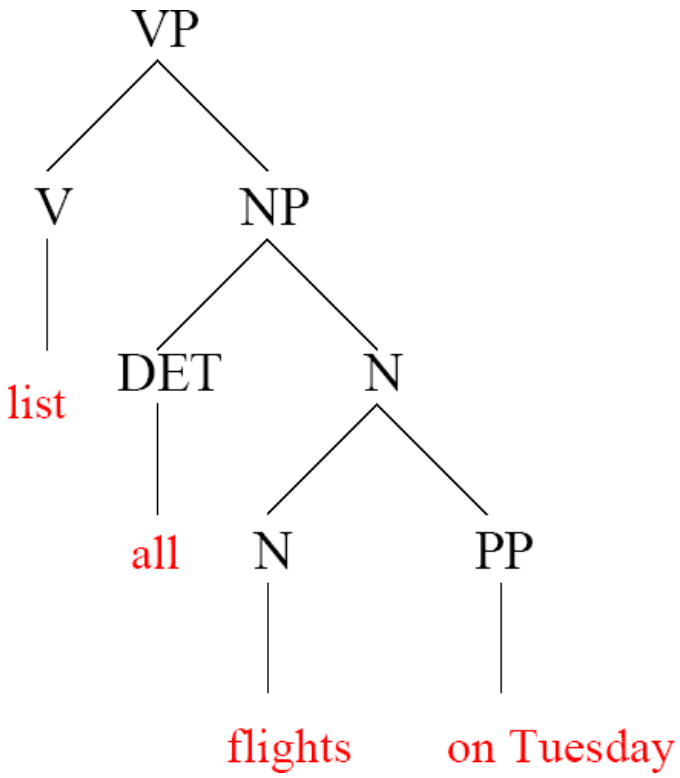
- At the syntactic level (structural ambiguity)
  - "(At last, computer) understand you like your mother"

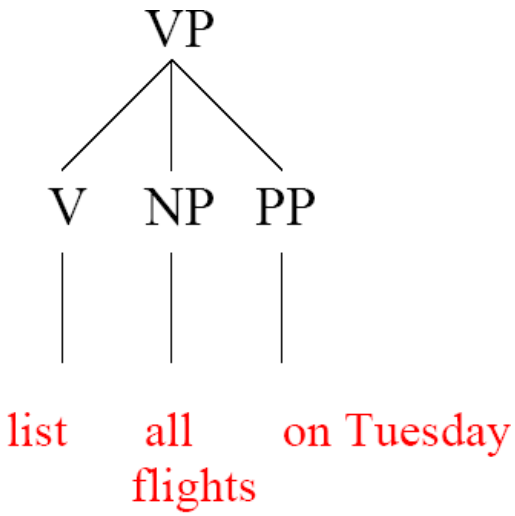


  - Different structures lead to different semantic interpretations

Structural Ambiguity

- One more example of structural ambiguity





Ambiguities at Multi Levels (3/3)

- At the semantic level (word sense ambiguity)
  - "(At last, computer) understand you like your *mother*"
  - The word 'Mother' has two different meanings in English
  - #1. A woman who has given birth to a child
  - #2. A stringy slimy substance consisting of yeast cells and bacteria; is adde d to cider or wine to produce vinegar (효모)

How to resolve ambiguity?

- We need
  - Knowledge about language (언어에 관한 지식)
    - For example, grammar, dictionary, …
  - Knowledge about the world (세상에 관한 지식)
    - Language acts are the product of agents in the world, either human or computer
    - For example, knowledge about the preference of a human, situation, …

Major Issues for NLP

- Thus, major issues for NLP would be
  - How to acquire necessary knowledge?
  - How to use such knowledge for resolving ambiguity?

- Two approaches
  - Symbolic approach (or rule-based approach, rational)
    - Built all necessary information by human's hand
    - Require excessive manual works (Is it possible to encode all necessary knowledge manually?)
  - Statistical approach (Empirical)
    - Try to automatically infer knowledge from samples (manually annotated corpus)

NLP applications

- Text Categorization
- Spelling & Grammar Corrections
- Information Extraction
- Speech Recognition
- Information Retrieval
- Summarization
- Machine Translation
- Question Answering
- Dialog Systems

NLP Basics

Preprocessing

Preprocessing (Error Correction)

- Problem domain
  - Correct all typographical errors in text before processing the given text
  - Previous research of error correction has focused on two types of errors :
    - Spacing error (띄어쓰기 오류)
      - “아버지가방에 들어 가셨다”
      - » “아버지가 방에 들어 가셨다.” vs. “아버지 가방에 들어 가셨다.”
  - Spelling error (철자오류)
    - “나는 하교에 갔다”
    - » “나는 학교에 갔다”
  - Both errors will significantly deteriorate performance of whole NLP system.
  - Complex errors (복합 오류) (e.g. “lemme C” => “Let me see.”)

Spacing Error Correction

- Two approaches for Korean Spacing error correction
  - Rule based approach
    - Based on Linguistic perspective
    - Computationally complex
    - A considerable cost for constructing and maintaining lexical inform
  - Statistical approach
    - Uses word-spacing probability estimated from every syllable bigram in the corpora
    - Statistically decide whether to insert a space between syllables or not

Spacing Error Correction

- Example of a simple rule based method for spacing error correction

들어+가셨다
➔들어+space+가셨다

아버지가방에 **들어가셨다**
➔ 아버지가방에 **들어 가셨다**

- However, there can be ambiguity between rules

가방 ➔가+space+방

non_space+가방
➔ space+가방

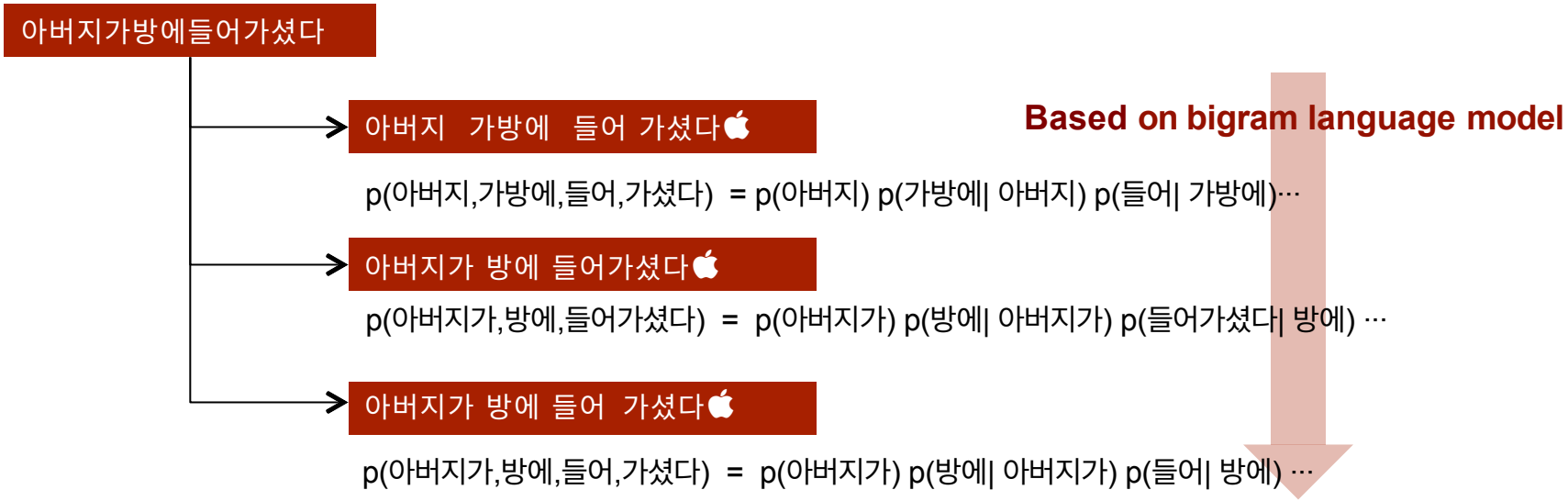
Which one is better?

아버지가방에 들어가셨다
➔ 아버지**가 방에** 들어 가셨다

아버지가방에 들어가셨다
➔ **아버지 가방에** 들어 가셨다

Spacing Error Correction

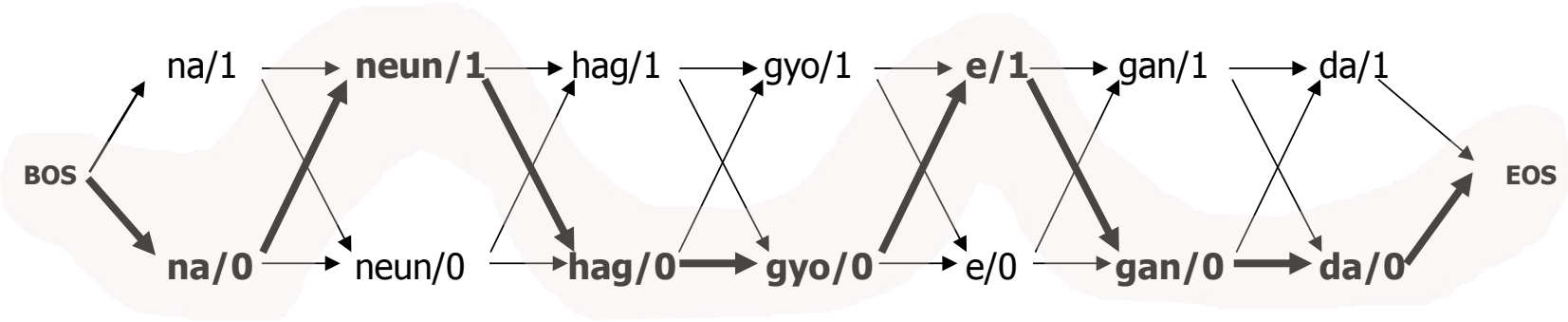
- Example of a simple statistical method
  - We can estimate likelihood for the possible correction candidates by using a language modeling method



In highly productive language such as Korean, data sparseness problem is a serious problem

Spacing Error Correction

- Alternative statistical method for Korean spacing error correction
  - When a sentence “나는 학교에 간다(na-neun hag-gyo-e gan da)” is given the result of spacing is



Spelling Error Correction (1/4)

- To correct spelling errors in text, two steps are required
  - 1. **Detect** possible erroneous strings
    - For example for the sentence "**Your** an idiot"
      - The word **"Your"** seems to be misspelled
  - 2. **Correct** errors
    - For example, "**Your** an idiot" → **You're** an idiot
  - The spelling errors can happen because of typographical errors (insertion, deletion, transposition), which accidentally **produce a real word**
    - Thus, there can be ambiguity for detecting errors!
    - Example : "Three are desert"
      - Can be misspelled one for "three are deserts", "there are deserts" or "three are desserts"

Spelling Error Correction (2/4)

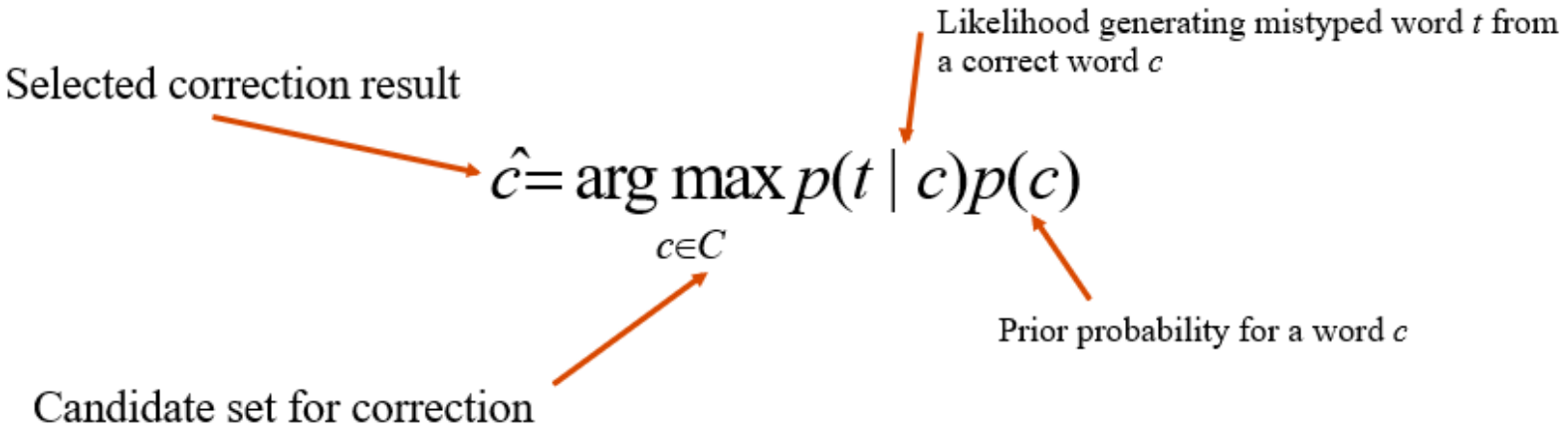
- Four types of single-error misspellings
  - Insertion : mistyping “the” as “ther”
  - Deletion : mistyping “the” as “th”
  - Substitution : mistyping “the” as “thw“
  - Transposition : mistyping “the” as “ hte”
- Example of ambiguities at spelling error correction task

Possible error correction results – which one is correct?

Error	Correction	Transformation			
		Correct Letter	Error Letter	Position (Letter #)	Type
acress	actress	t	–	2	deletion
acress	cress	–	a	0	insertion
acress	caress	ca	ac	0	transposition
acress	access	c	r	2	substitution
acress	across	o	e	3	substitution
acress	acres	–	2	5	insertion
acress	acres	–	2	4	insertion

Spelling Error Correction (3/4)

- Simple statistical method to correct spelling errors
  - Bayesian inference model



Spelling Error Correction (4/4)

- For the example “actress”

$$\hat{c} = \arg\max_{c \in C} p(t \mid c)\, p(c)$$

c	freq(c)	p(c)	p(t|c)	p(t|c)p(c)	%
actress	1343	.0000315	.000117	3.69×10^{-9}	**37%**
cress	0	.000000014	.00000144	2.02×10^{-14}	**0%**
caress	4	.0000001	.00000164	1.64×10^{-13}	**0%**
access	2280	.000058	.000000209	1.21×10^{-11}	**0%**
across	8436	.00019	.0000093	1.77×10^{-9}	**18%**
acres	2879	.000065	.0000321	2.09×10^{-9}	**21%**
acres	2879	.000065	.0000342	2.22×10^{-9}	**23%**

NLP Basics

Morphological Level

Lexical Analysis

The need for lexical analysis

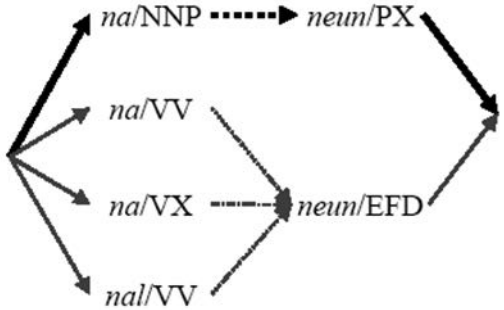
- To improve the efficiency by dividing the input sentences well

Key factors in lexical analysis

- Sentence splitting : period(.), exclamation mark(!), question mark(?)
- Tokenizing : splitting documents and sentences for analysis (In spacing or morpheme)
- Morphological : analyze tokens in more general form to reduce the number of words and increase the efficiency of the analysis (Tokenized to the smallest semantic unit)
- Stemming (‘cars’,’car’=>’car’), lemmatization (change word to original form),…

Morphological Analysis

- Morphological analysis means outputting all possible analysis results of an input word
  - Potential parts-of-speech for a given word (for English)
  - Or morphological parses for a given Eojeol (for Korean)
- Morphological analyzer should produce all the grammatically possible interpretations for a give word (or Eojeol)
- Example of morphological analysis in Korean
  - When a word “나는”(na-neun) is given, the result of morphological analysis is : (나 : 대명사, 싹이 나는, 새가 나는, …)



Morphological Analysis

- Two approaches for Korean Morphological Analysis
  - Rule and dictionary based Approaches
    - Depend on manually constructed linguistic resources such as
      - Morpheme dictionary, morpho-syntactic rules, morphological rules…
    - Try to find all valid morpheme sequences with constraint features and morpheme restoration rules
    - Fast but requires a lot of labor intensive knowledge
  - Statistical approach
    - Try to learn linguistic knowledge from POS tagged-corpus in a statistical manner
    - Does not require any morpheme dictionary and rules
    - Slow

Morphological Analysis

- Morphological Analysis
  - To understand the structure of various linguistic attributes such as morpheme, root, prefix / suffix, part-of-speech
- Application Model
  - Mecab, Twiiter, Komoran, Hannanum, KoNLPy 등

```
pprint(kkma.pos(a))
```

```
문장을 입력하세요:세종대왕님은 글을 만드셨습니다.
[('세종', 'NNG'),
 ('대왕', 'NNG'),
 ('님', 'XSN'),
 ('은', 'JX'),
 ('글', 'NNG'),
 ('을', 'JKO'),
 ('만들', 'VV'),
 ('시', 'EPH'),
 ('었', 'EPT'),
 ('습니다', 'EFN'),
 ('.', 'SF')]
```

Using KoNLPy Model

POS Tagging

- POS Tagging is short for Part-of-Speech Tagging
  - The task of determining the correct part-of-speech by eliminating the morphological ambiguity of words (phrases, morphemes)
  - Preprocessing process to reduce excessive burden in parsing phase
  - POS Tagging uses Rule-based Algorithm and Stochastic Algorithm.

가방에 들어가신다
-> 가방/**NNG** + 에/**JKM** + 들어가/**VV** + 시/**EPH** + ㄴ다/**EFN**

POS Tagging

- POS Tagging Considerations
  - There is a problem of processing words (morphemes) not registered in advance
  - Data Sparseness

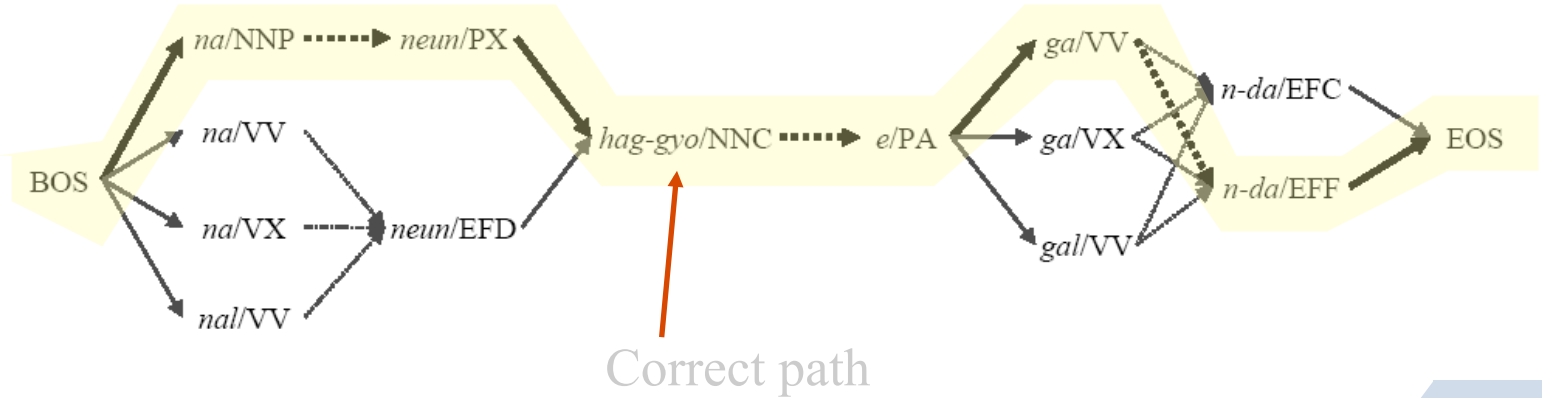
```
pprint(kkma.pos(a))
```

```
문장을 입력하세요:세종대왕님은 글을 만드셨습니다.
[('세종', 'NNG'),
 ('대왕', 'NNG'),
 ('님', 'XSN'),
 ('은', 'JX'),
 ('글', 'NNG'),
 ('을', 'JKO'),
 ('만들', 'VV'),
 ('시', 'EPH'),
 ('었', 'EPT'),
 ('습니다', 'EFN'),
 ('.', 'SF')]
```

Using KoNLPy Model

Statistical POS Tagging (1/9)

- Part of Speech (POS) tagging is
  - A task to assign a proper POS tag to each linguistic unit such as a word (in English), or a morpheme (in Korean) for a given sentence
    - An input of POS tagger is a morphological analysis result, and an output is a correct sequence of morpheme-POS pairs



Statistical POS Tagging (2/9)

- Hidden Markov Model (HMM) based POS Tagging
  - Most popular and well-performed approach
    - Regard POS tags of morphemes in a given sentence as hidden states

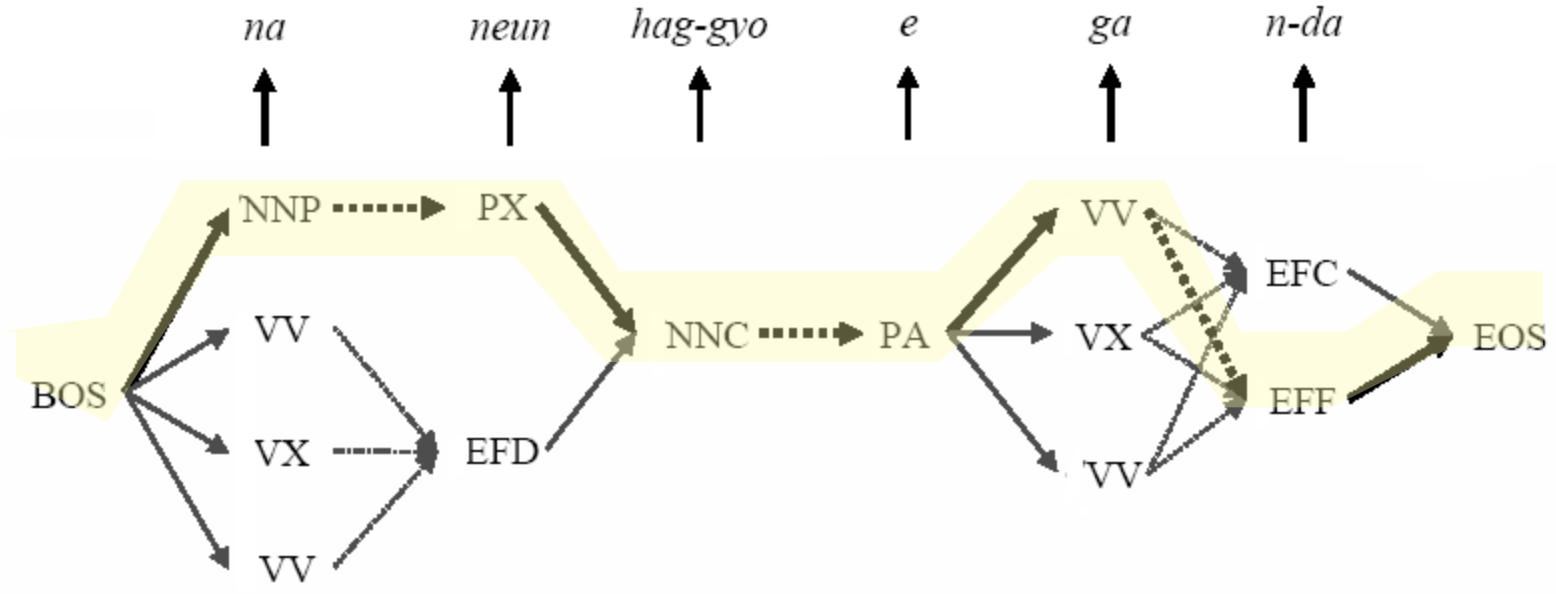


Figure: HMM representation for POS tagging problem

Statistical POS Tagging (3/9)♪

- Example of a simple HMM method for POS tagging

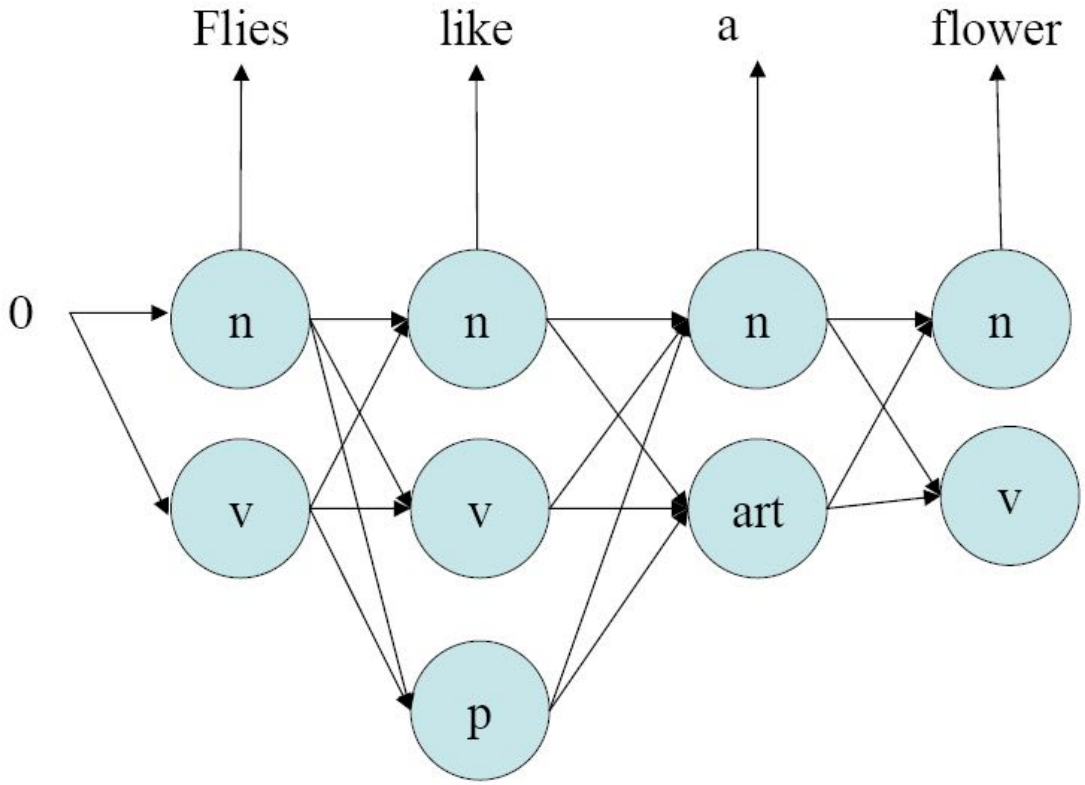
$$\mathrm{T}(w_{1,N}) \overset{def}{=} \arg\max_{c_{1,N}} \prod_{i=1}^{N} P(c_i \mid c_{i-1}) P(w_i \mid c_i)$$

Emission Probability

Transition Probability

Statistical POS Tagging (4/9)♪

- Analyze the sentence "*Flies like a flower*"♪

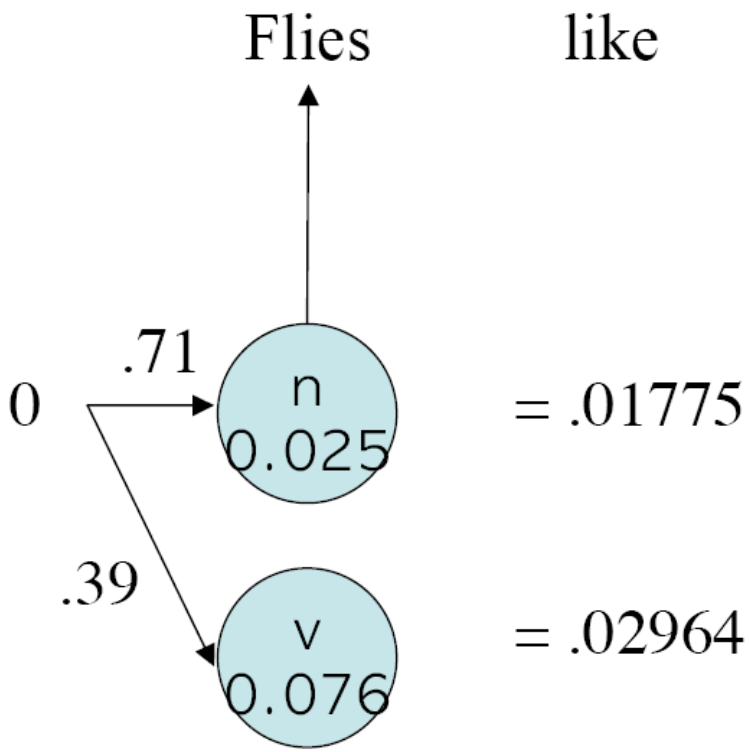


Statistical POS Tagging (5/9)♪

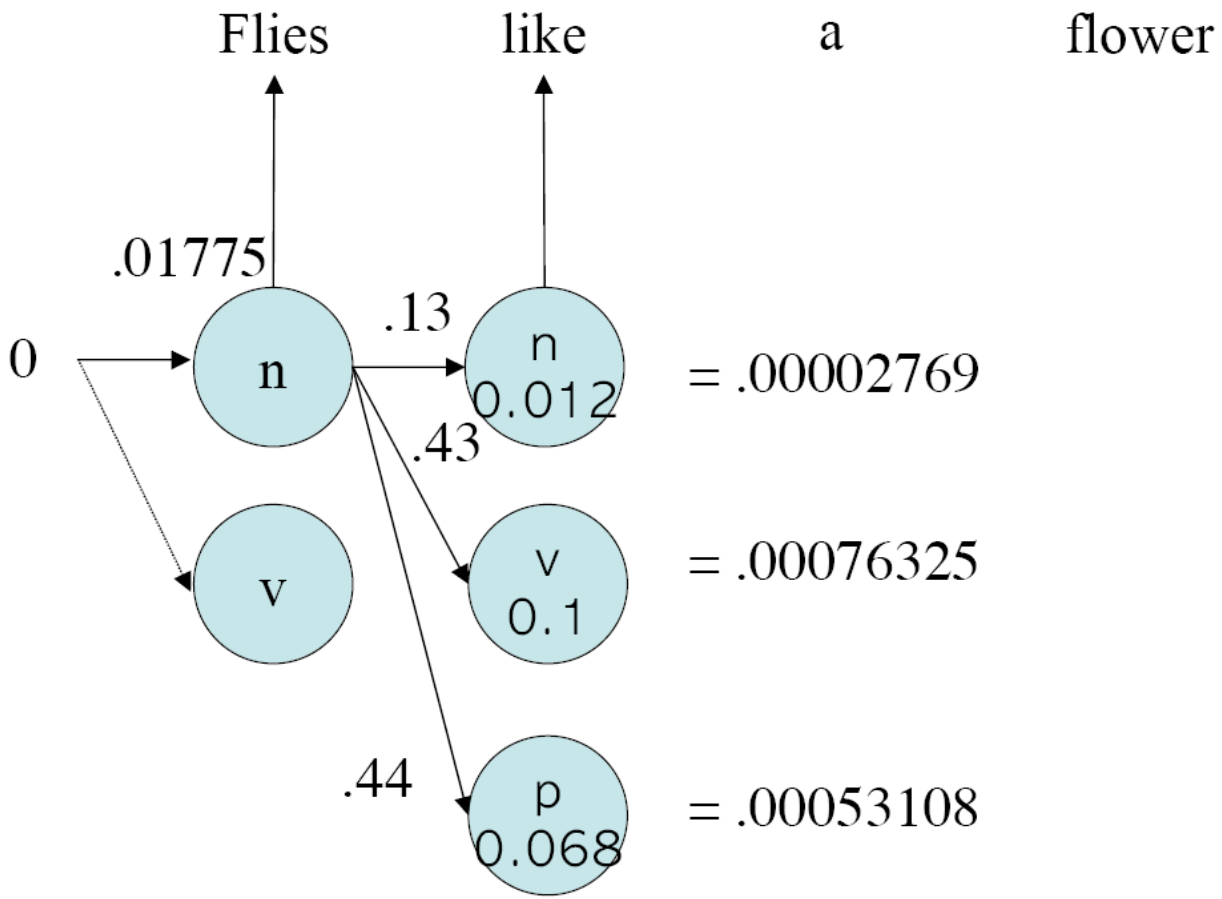
- Statistics♪

품사	#i	품사쌍	#i,i+1	Bigram	확률
ㅇ	300	ㅇ, ART	213	p(ART|ㅇ)	.71
ㅇ	300	ㅇ,N	87	p(N|ㅇ)	.29
ART	558	ART,N	558	p(N|ART)	1
N	833	N,V	358	p(V|N)	.43
N	833	N,N	108	p(N|N)	.13
N	833	N,P	366	p(P|N)	.44
V	300	V,N	75	p(N|V)	.35
V	300	V,ART	194	p(ART|V)	.65
P	307	P,ART	226	p(ART|P)	.74
P	307	P,N	81	p(N|P)	.26

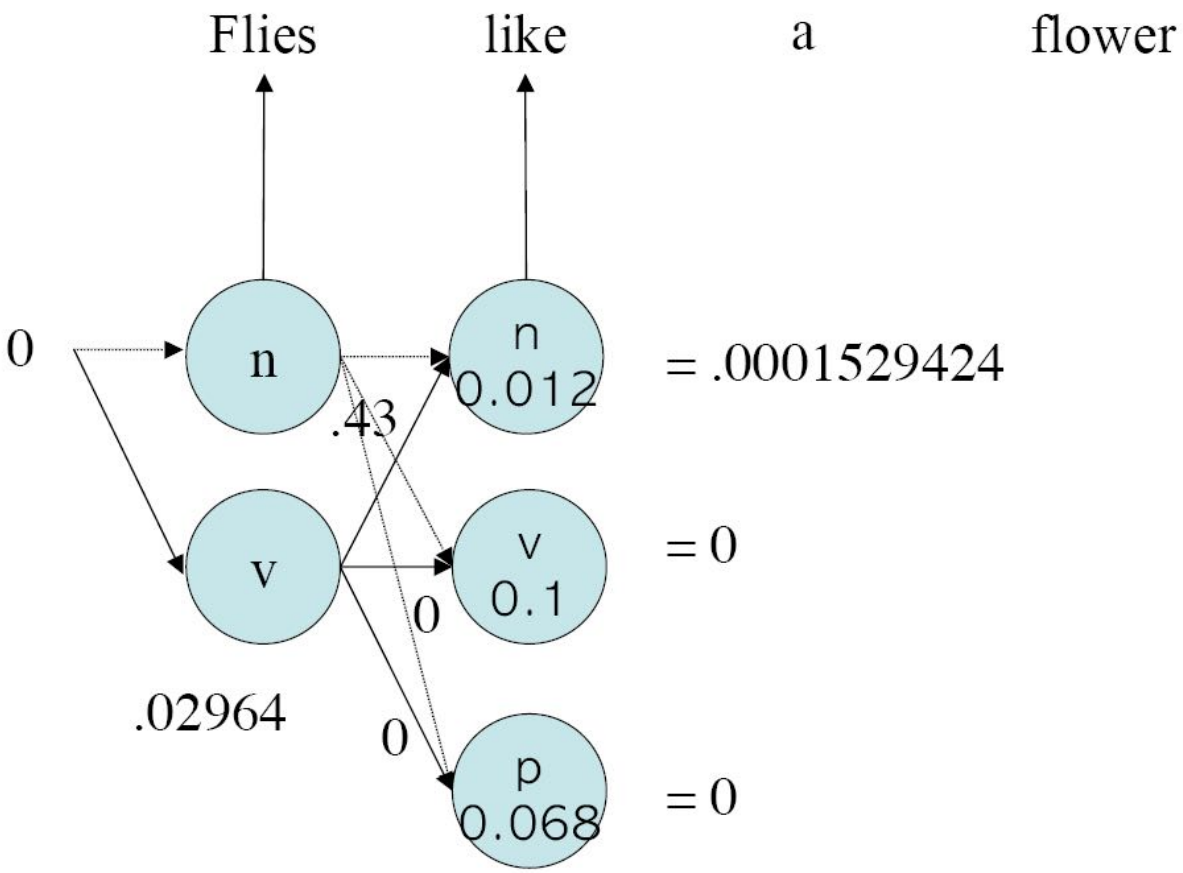
Statistical POS Tagging (6/9)♪



Statistical POS Tagging (7/9)♪

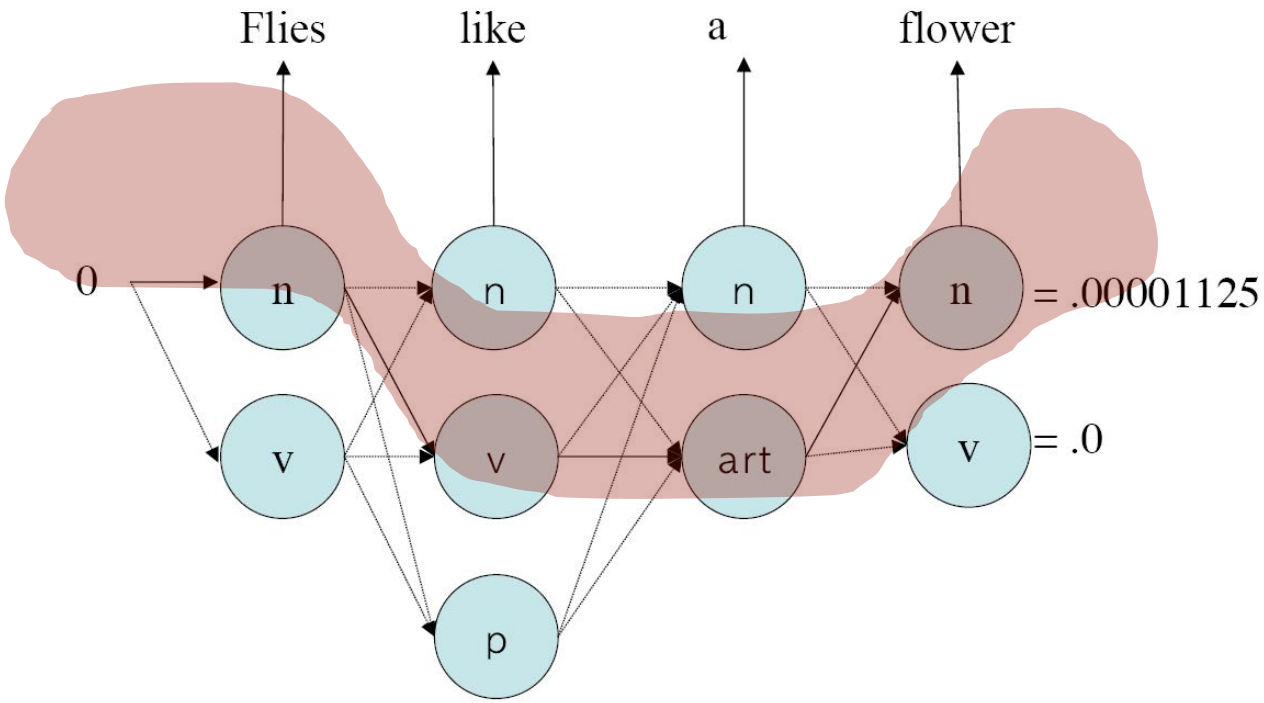


Statistical POS Tagging (8/9)♪

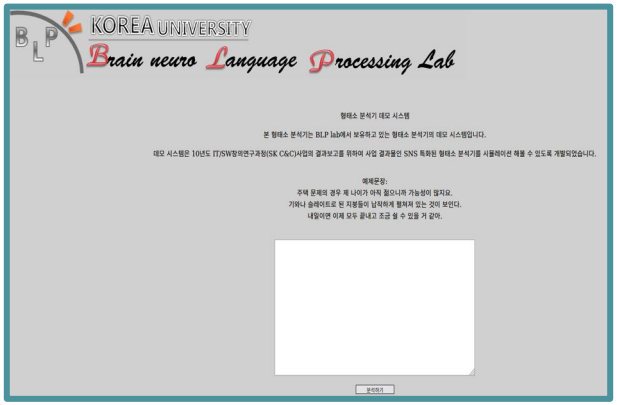


Statistical POS Tagging (9/9)♪

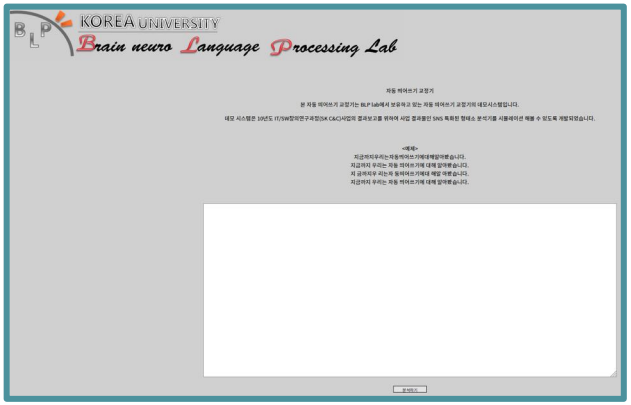
- Finally, we can find the POS sequence maximizing prob.♪



Morphological analyzer / automatic spacing braces



Morphological analyzer



Automatic spacing corrector

Morphological analyzer
http://blpdemo.korea.ac.kr/MA/
Morphological analyzer using statistical model

Automatic spacing corrector
http://blpdemo.korea.ac.kr/autospacing/
Automatic spacing corrector using statistical model

Named Entity Recognition

- NER(Named Entity Recognition)
  - It is a sub-task of information extraction by classifying object names into predefined categories such as personal, organization, and company name
  - Most of the studies on NER systems such as location, time representation, quantity, organs are annotated

NE Type	Examples
ORGANIZATION	*Georgia-Pacific Corp., WHO*
PERSON	*Eddy Bonte, President Obama*
LOCATION	*Murray River, Mount Everest*
DATE	*June, 2008-06-29*
TIME	*two fifty a m, 1:30 p.m.*
MONEY	*175 million Canadian Dollars, GBP 10.40*
PERCENT	*twenty pct, 18.75 %*
FACILITY	*Washington Monument, Stonehenge*
GPE	*South East Asia, Midlothian*

Named Entity Recognition

- NER Goal : Identify all Named Entities (NEs)

1. Finding NEs
2. Identify the type of NE found

- When performing text mining in a specific field, it is better to learn Tagger and Recognizer by using a corpus suitable for it

Jim bought 300 shares of Acme Corp. in 2006.

-> Jim bought 300 shares of Acme Corp. in 2006.

Person Organization Time

NLP Basics

Syntactic Level

Syntax analysis

The need for syntax analysis

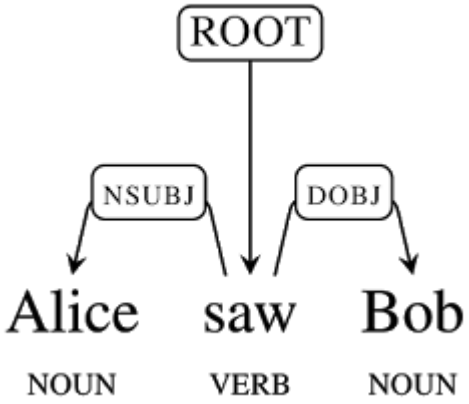
- Language requires rules to construct sentences.
- Construct rules/grammar for constructing sentences.

What is syntax analysis (syntactic) ?

- Separate by each word unit, and then grant tag (using parsing tree).
- Represented as dependency parsing tree (dependency parsing).
- Alice (subject) and Bob (object) relationship based on Saw (root).
- These grammatical relations are very directly related.

POS	UMLS	Penn Tree Bank Tag	Example
Noun	noun	NN, NNS, NNP, NNPS	table
Adjective	adj	JJ, JJR, JJS	blue
Adverb	adv	RB, RBR, RBS, WRB, RP	quickly
Pronoun	pron	PRP, PRP$, WP, WP$	she
Verb	verb	VB, VBD, VBG, VBN, VBP, VBZ	wrote
Determiner	det	DT, PDT, WDT	the
Preposition	prep	IN	with
Conjunction	conj	CC	and
Auxiliary	aux	VB, VBD, VBG, VBN, VBP, VBZ	does
Modal	modal	MD	could
Complement	compl	IN	that

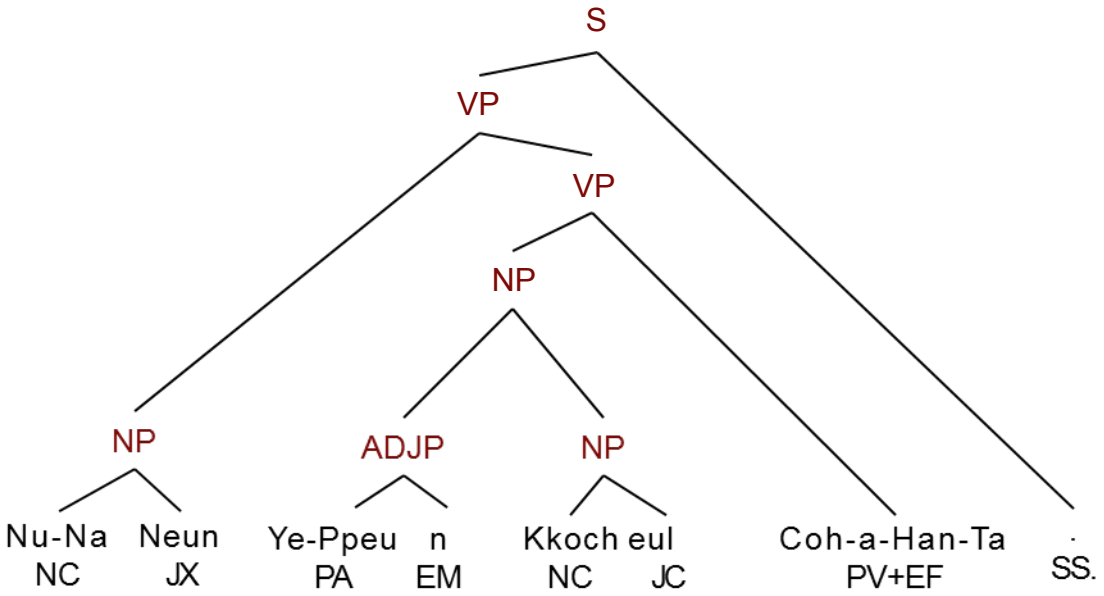
Part-of-speech tagging



Dependency parsing

Syntactic Parsing

- Goal
  - Find out the syntactic structure with a specific grammar for a given sentence
    - Example of pared sentence with the phrasal structure grammar
      - “누나는 예쁜 꽃을 좋아한다. (Nu-na-Neun ye-Ppeun Kkoch-eul Coh-a-Han-Ta)”



Syntactic Parsing

- Parsing can be defined as
  - A problem that maps any input sentence to an appropriate syntactic tree structure
  - Why is the parsing so difficult?
    - Simply, the answer is ***structural ambiguity of natural language !!***
    - Especially, several characteristics of Korean make the parsing more difficult
      - Relatively free-word order, discontinuous constituents, constituent ellipsis …

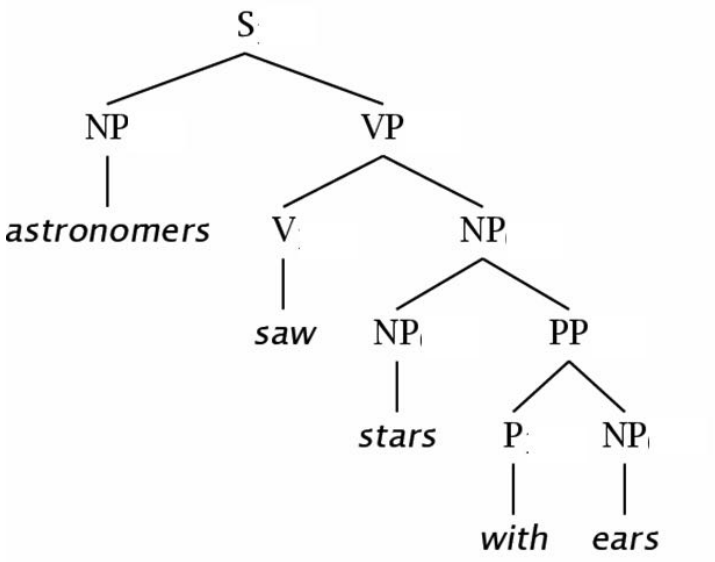
Syntactic Parsing

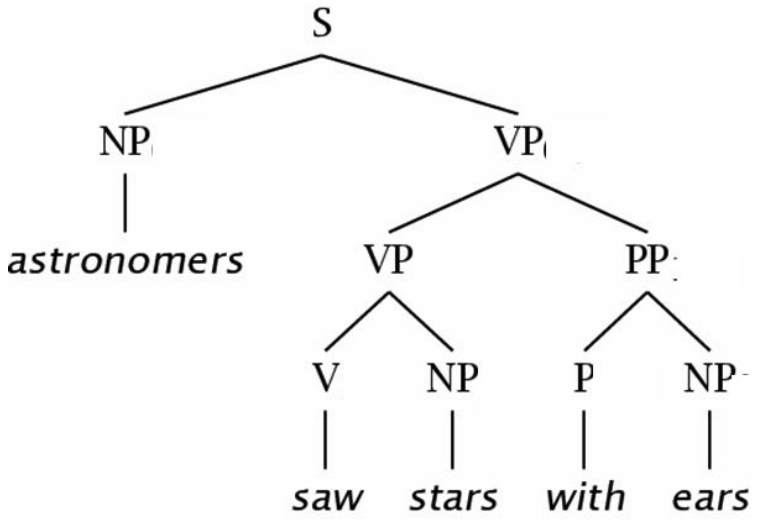
- Two approaches for syntactic parsing
  - Rule-based approach
    - Try to resolve structural ambiguities of natural language with rich linguistic knowledge such as sophisticated grammar
    - Mainly, the machine performs disambiguation by using the selection restrictions
    - Several Problem : Incredible amount of manual labor and all-or-nothing nature of selection restriction
  - Statistical approach
    - Try to find the most probable syntactic tree for a given sentence
      - Does not require any hand-crafted grammar but the annotated corpus
      - A parser is trained and learned linguistic preferences from the annotated corpus, completely automatically

Syntactic Parsing

- Example of statistical parsing method based on PCFG
  - The sentence " Astronomers saw stars with ears" can be analyzed in two ways

How to decide which one is correct?





Syntactic Parsing

- Infer PCFG grammar from Tree bank

S → NP VP	1.0	NP → NP PP	0.4
VP → V NP	0.7	NP → *astronomers*	0.1
VP → V NP PP	0.3	NP → *ears*	0.18
PP → P NP	1.0	NP → *saw*	0.04
P → *with*	1.0	NP → *stars*	0.18
V → *saw*	1.0	NP → *telescope*	0.1

Example of treebank

```
; 사람이 스스로 만물의 영장이라 하고 우쭐대는 까닭이 여기에 있다.
(S
  (ADJP
    (NP
      (VP (NP 사람 )+이
        (VP
          (VP (ADVP 스스로 )
            (VP
              (VP
                (NP (NP 만물 )+의
                  영장 )+이 )+라
                하 ))+고
              우쭐대 ))+는 까닭 )+이
    (ADJP (NP 여기 )+에 있 ))+다+. )
```

Syntactic Parsing

- Simple Parsing Model based on PCFG

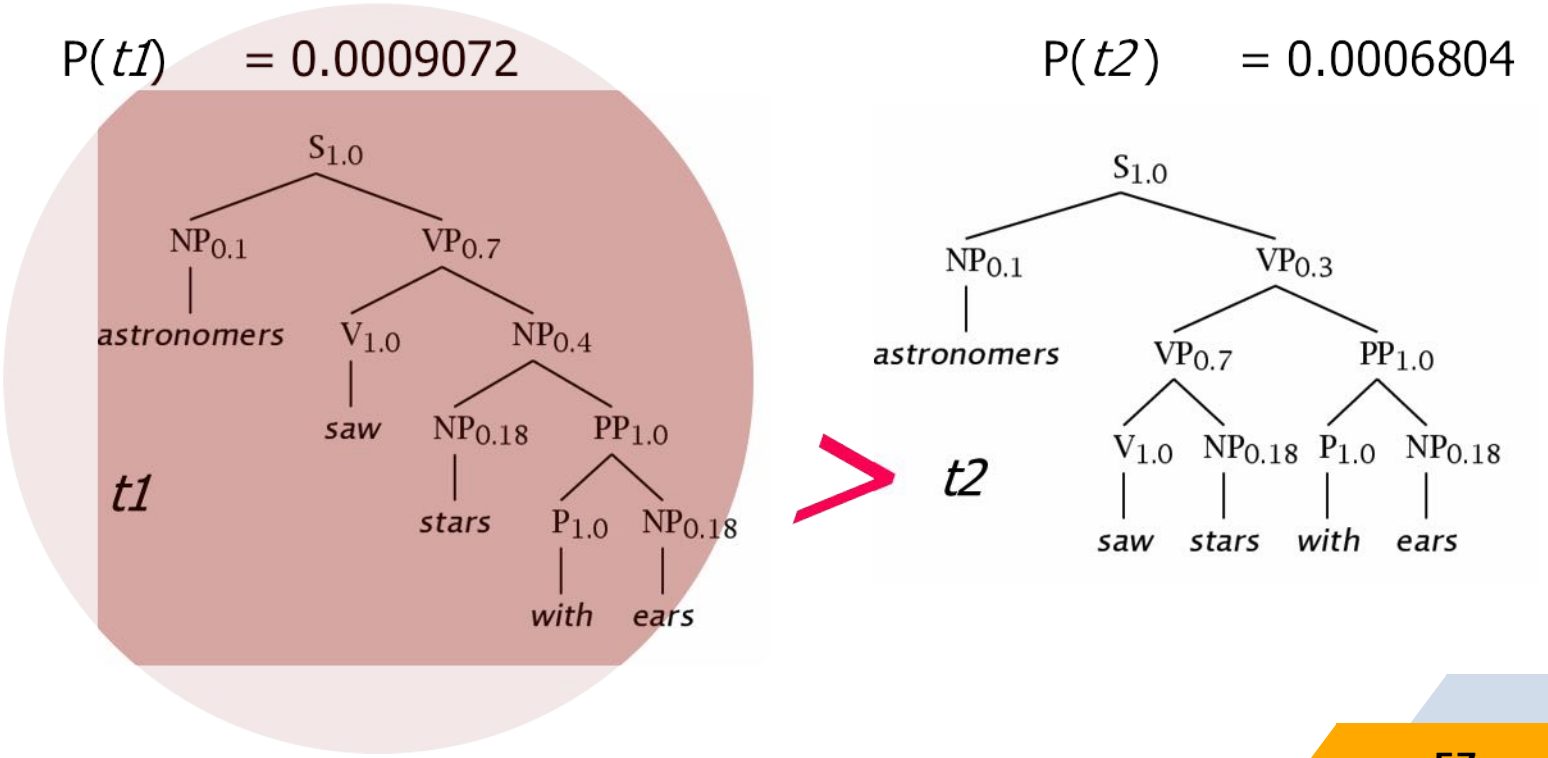
$$\hat{T}(S) = \arg\max_{T \in \tau(S)} P(T)$$

$$= \arg\max_{T \in \tau(S)} \prod_{n \in T} p(r(n))$$

Probability of Rule $r(n)$

Syntactic Parsing

- Finally, we can know which tree is more likely



NLP Basics

Semantic Level

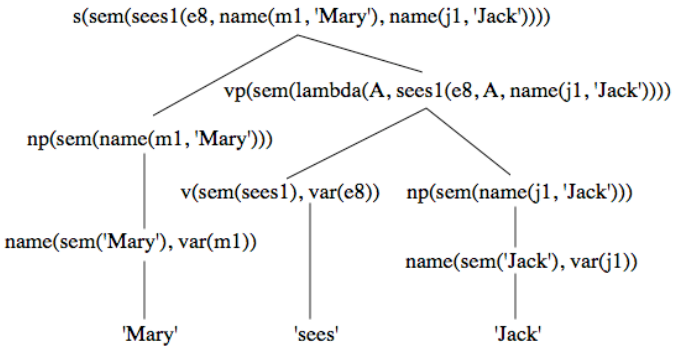
Semantic analysis

The need for Semantic analysis

- In the previous process (i.e. syntax analysis), we have created a sentence according to the rules and need to know if the sentence is semantically correct.
- A person eats an apple. (O)
- A person eats an airplane. (X)

What is semantic analysis?

- Syntactic + meaning
- Two levels (lexical semantics)
- Representing meaning of words
- Word sense disambiguation (word bank)



Syntax & semantics

- Compositional semantics
- How words combined to form a larger meaning
- Sentence - A tall man likes Mary
- Representation -x man(x) & tall(x) & likes (x, mary)

Semantic analysis

- Semantic analysis is
  - The process whereby meaning representations are composed and assigned to natural language text (의미표현이 구성되고 할당되는 처리)

Word Sense Disambiguation

- Getting computers to find the correct meaning of a word in context
- Ex) What sort of **plants** thrive in chalky soil?

Plant ?

Plant ?

- Applications
- Machine translation, Information Retrieval, QA, speech synthesis

Word Sense Disambiguation Approaches

- supervised (hand labelled data)
- knowledge-based (dictionaries, thesaurus)
- unsupervised
  - induce senses (fully unsupervised) similarity of input vector to previous clusters (LSA)
  - or associate distributional information with entries in given sense inventory NB association uses knowledge

Semantic Role Labeling

- SRL stands for Semantic Role Labeling
- SRL is Classify semantic discovery and specific roles associated with predicates or verbs in sentences
- SRL is useful as an intermediate step in a variety of NLP operations
  - such as information extraction, automatic document classification, and question answering

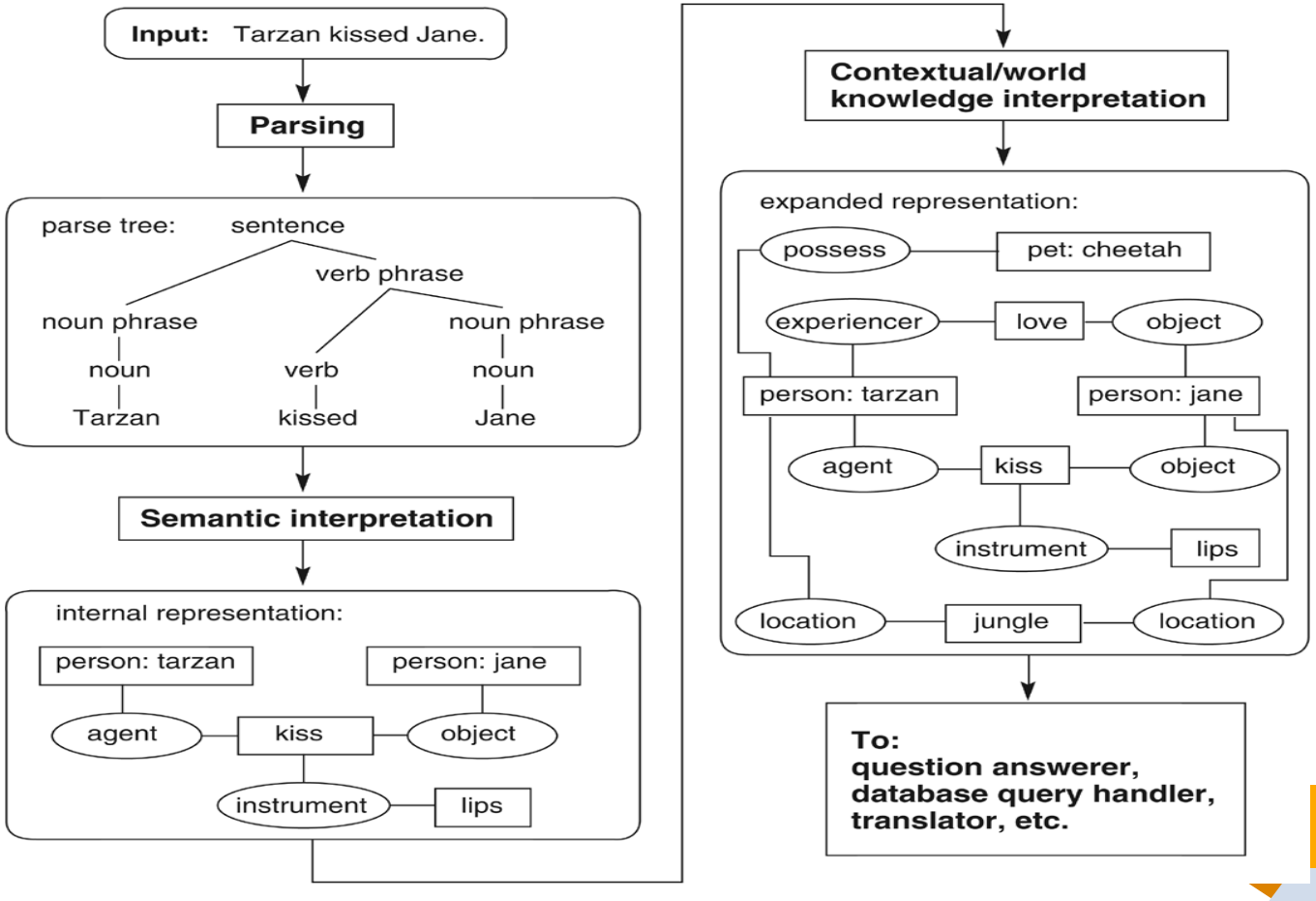
Semantic Role Labeling

The Police officer	detained	the suspect	at the scene of the crime
Agent	Predicate	Theme	Location

Thematic Role	Definition
AGENT	The volitional causer of an event
EXPERIENCER	The experiencer of an event
FORCE	The non-volitional causer of the event
THEME	The participant most directly affected by an event
RESULT	The end product of an event
CONTENT	The proposition or content of a propositional event
INSTRUMENT	An instrument used in an event
BENEFICIARY	The beneficiary of an event
SOURCE	The origin of the object of a transfer event
GOAL	The destination of an object of a transfer event

Example of Semantic analysis

- Example of semantic analysis



NLP Basics

Discourse Level

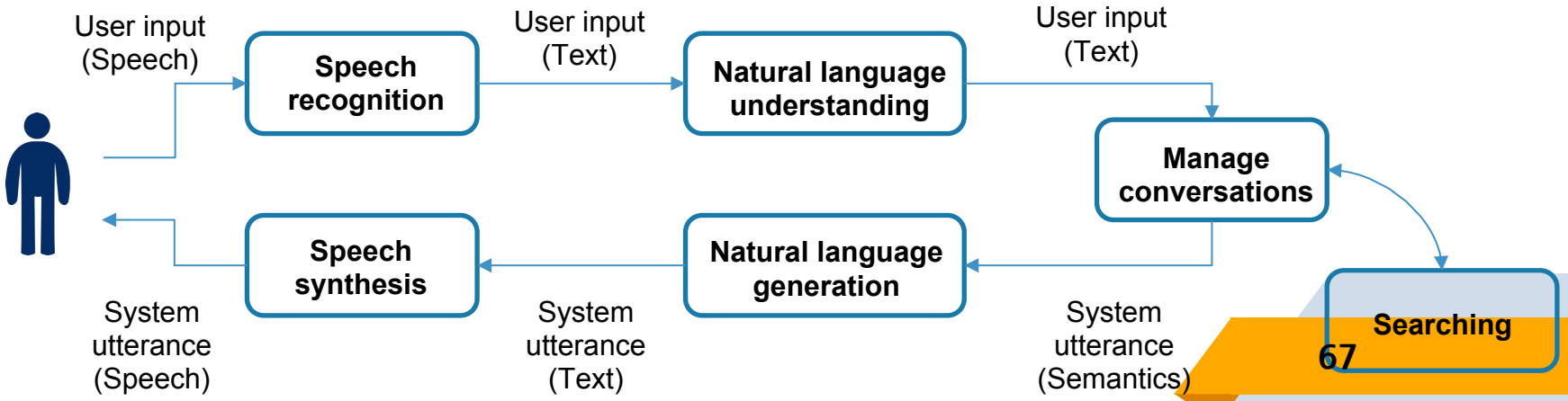
Discourse analysis

The need for discourse analysis

- Discourse analysis is needed to understand the flow of conversation and respond to the speaker's intentions.

What is discourse analysis?

- Looks for what the meaning of the dialogue flow is
- Context structure analysis (specially, Relationship between sentences)
- Intention analysis (understand actual intentions through context)
- Dialogue analysis (representative discourse analysis)



Dialogue system process

Discourse Analysis

- Reference resolution (대용어 해결)
  - The omitted words (or phrases) and the pronominal references are complemented by the use of common sense and discourse information
- Speech Act Identification (화행분석)
  - Speech Act: The communicative intention represented by each utterance (진술, 주장, 추측, 명령, 요청, 언약, 정표 등) 발화의 의도
  - A dialog system should have the ability to
    - identify other participants' speech act, predict next possible speech acts, and generate own utterance suitable for the speech act

U: I would like to open a fixed deposit account.
S: For what amount?
U: Make it for 8000 dollars.

NLP Applications

Sentiment Analysis

- It refers to the analysis of extracting subjective impressions, attitudes, and opinions of individuals on texts from texts
- Generally, a binary opposition in opinions is assumed
  - For/against, like/dislike, good/bad, etc.
  - Some sentiment analysis jargon:
    - “Semantic orientation”
    - “Polarity”

Sentiment Analysis

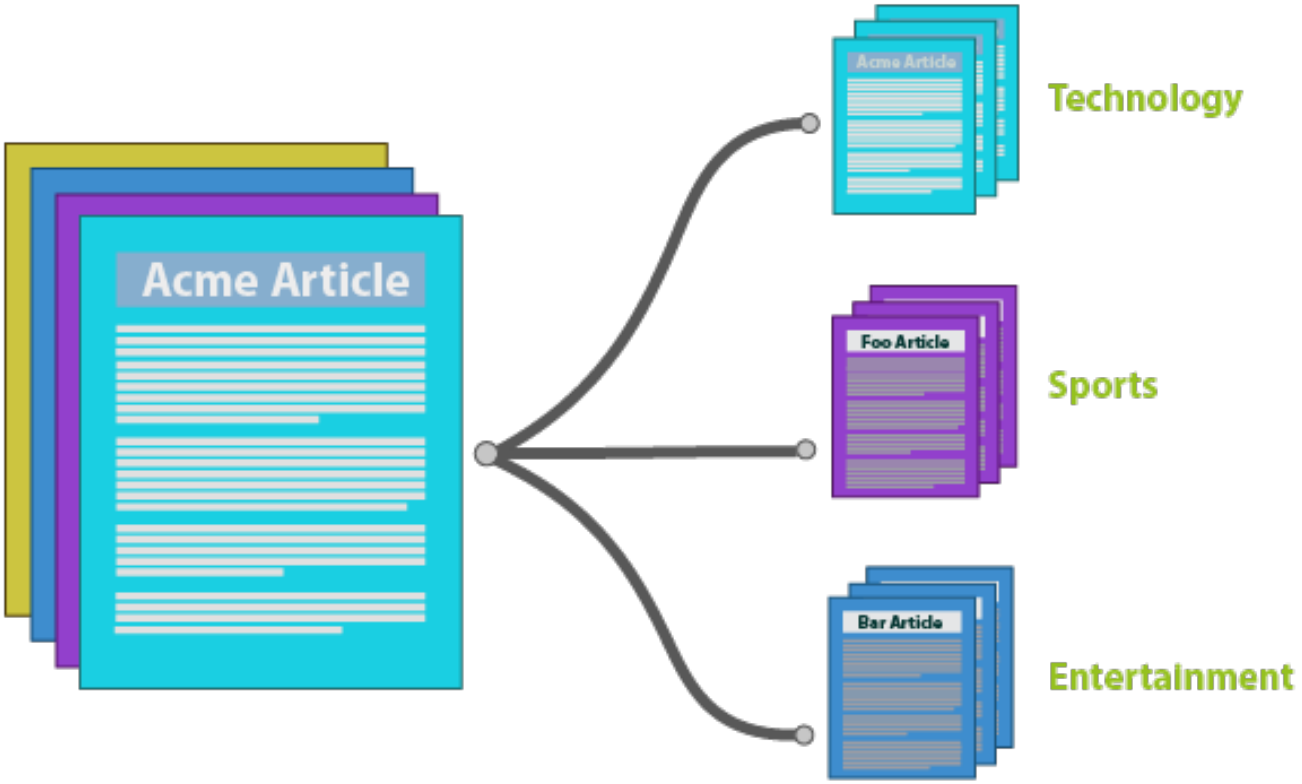
- It is based on grasping affirmative, negative or neutral in a given text, and it finds the polarity of the text
- It mainly uses SVM (Support Vector Machines) and Logistic Regression classification algorithm.

Find people's opinions and feelings about your products or services

Text Classification

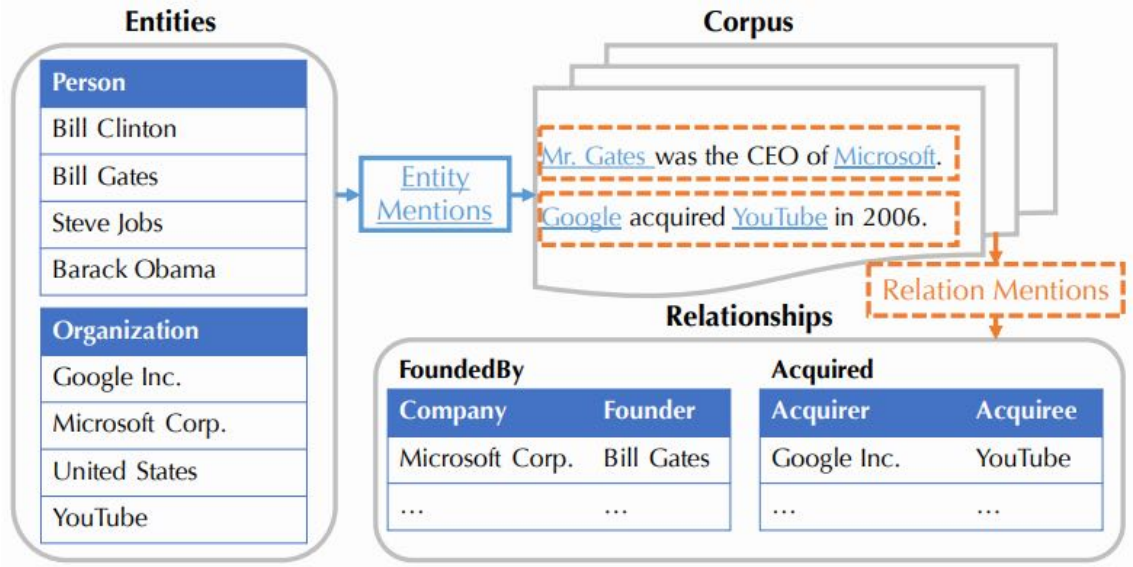
- It is classified as one or more classes or categories depending on contents
- In the past, document classification was performed manually, but in this case, it was difficult for people in terms of effort, time, and cost

Text Classification



Knowledge Extraction

- the creation of knowledge from structured and unstructured (text, documents, images) sources
- The resulting knowledge needs to be in a machine-readable and machine-interpretable format and must represent knowledge in a manner that facilitates inferencing



Knowledge Extraction

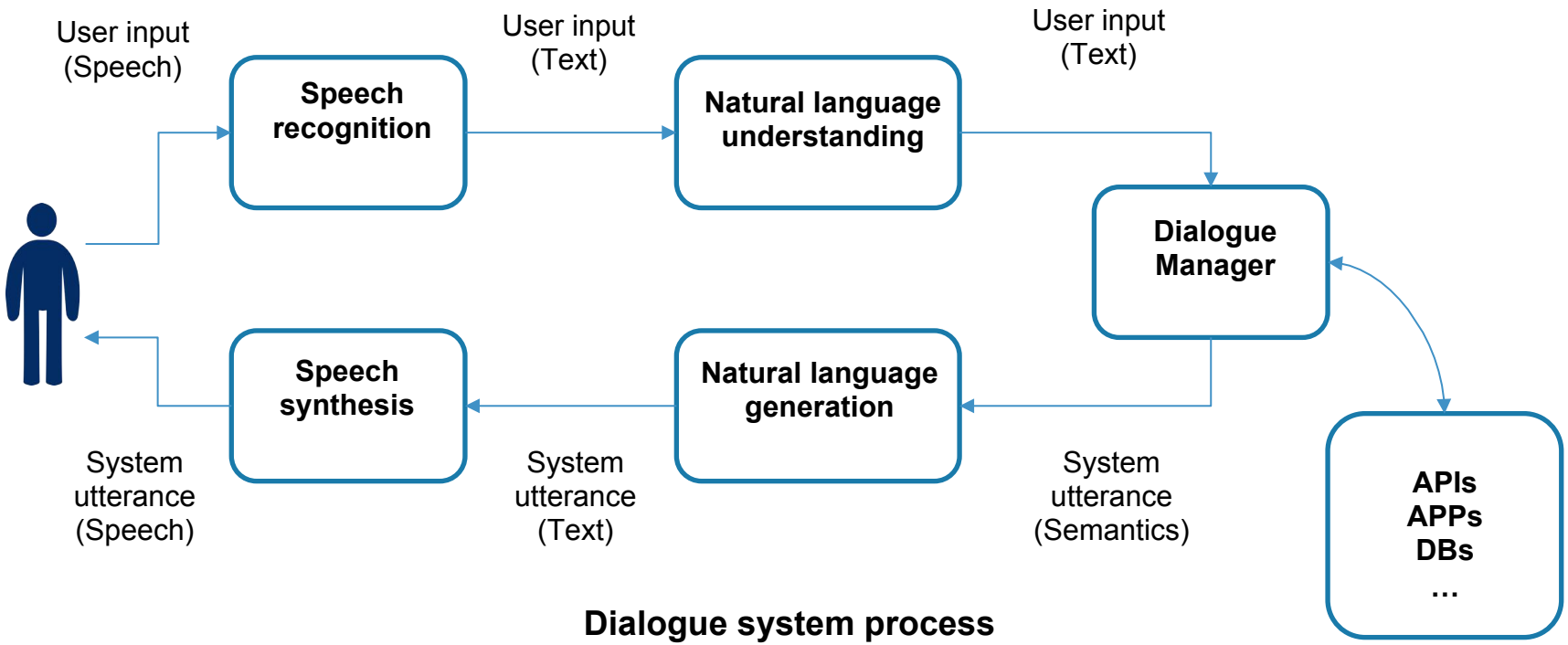
Raw text	Structured facts
Marie Curie was born on November 1867, 7. She was a Polish and naturalized-French physicist and chemist who conducted pioneering research on radioactivity...	(Marie Curie, date_of_birth, 11/07/1867)
	(Marie Curie, nationality_at_birth, Polish)
	(Marie Curie, nationality_at_death, French)
	(Marie Curie, job, physicist)
	(Marie Curie, job, chemist)
	(Marie Curie, job, researcher)
	(Marie Curie, research_field, radiocativity)
	...

- The 'Raw text' on the left contains a lot of useful information in an unstructured way, such as birthday, nationality, activity
- Extracting this information may require statement parsing, entity detection, etc. that aggregate information about the same entity
- In summary, Knowledge Extraction:
  - ✓ It can make a query and then get the requested information
  - ✓ It is to perform arbitrarily complex reasoning by finding paths in a graph of extracted knowledge

Dialogue System

- A Dialog System is a computer system intended to converse with a human, with a coherent structure
- Dialog systems are required to be applied to the system by users with very limited vocabulary and themes
- Application
  - It can be applied to a wide range of fields such as call centers, enterprises, and education.

Dialogue System



Dialogue system process

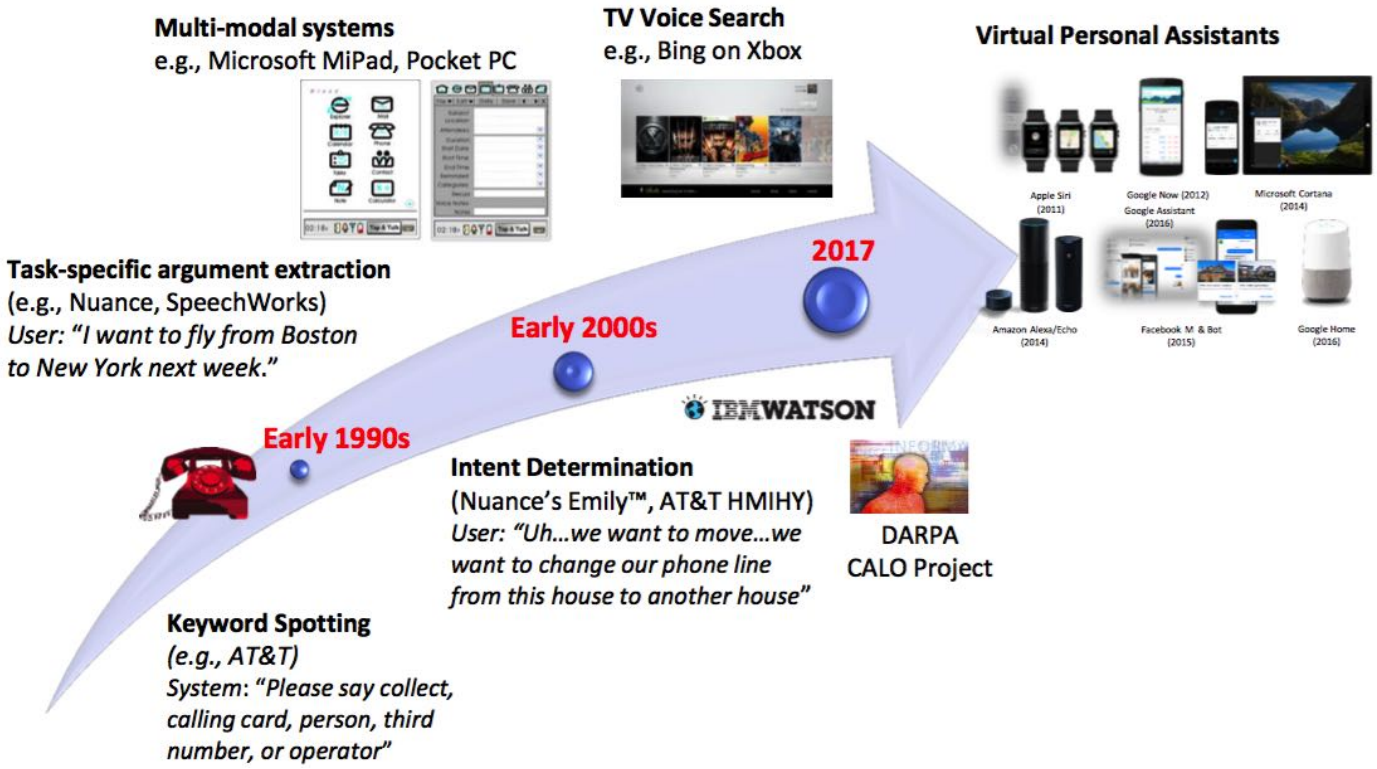
Dialogue System

- A typical activity cycle in a dialog system contains the following phases:

1. The user speaks, and the input is converted to plain text by the system's input recognizer/decoder, which may include
2. The text is analyzed by a Natural language understanding unit (NLU), which may include
3. The semantic information is analyzed by the dialog manager, that keeps the history and state of the dialog and manages the general flow of the conversation
4. Usually, the dialog manager contacts one or more task managers, that have knowledge of the specific task domain
5. The dialog manager produces output using an output generator, which may include
6. Finally, the output is rendered using an output renderer, which may include

Dialogue System

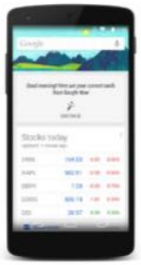
History of Dialogue Dialogue Systems



Dialogue System

Intelligent Assistant

Apple Siri (2011)



Google Now (2012)
Google Assistant (2016)

Microsoft Cortana (2014)

Amazon Alexa/Echo (2014)

Facebook M & Bot (2015)

Google Home (2016)

Apple HomePod (2017)

Dialogue System

Why We Need?

- Get things done
  - Set up alarm/reminder, take note
- Easy access to structured data, services and apps
  - Find docs/photos/restaurants
- Assist your daily schedule and routine
  - Commute alerts to/from work
- Be more productive in managing your work and personal life

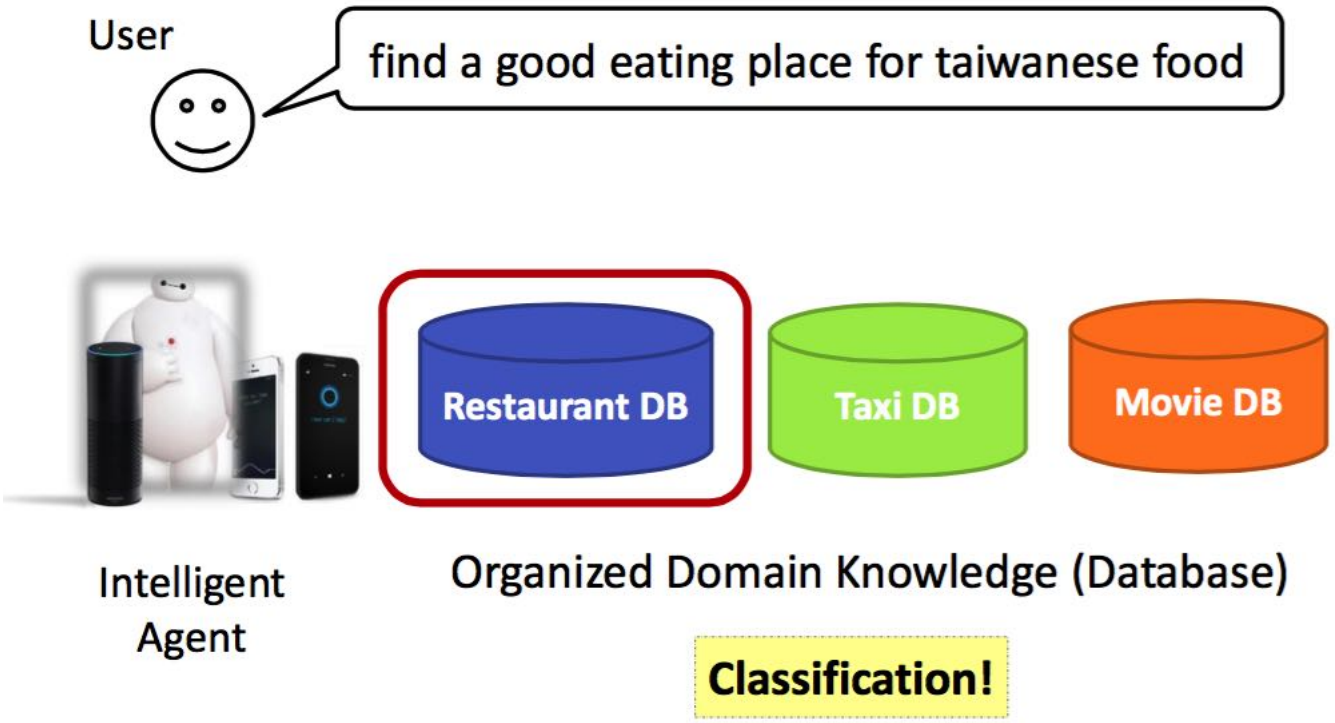
Dialogue System

Spoken Dialogue System (SDS)

- Spoken dialogue systems are intelligent agents that are able to help users finish taks more efficiently via spoken interactions.
- Spoken dialogue systems are being incorporated into various devices (smart-phones, smart TVs, etc)

Dialogue System

Domain Identification

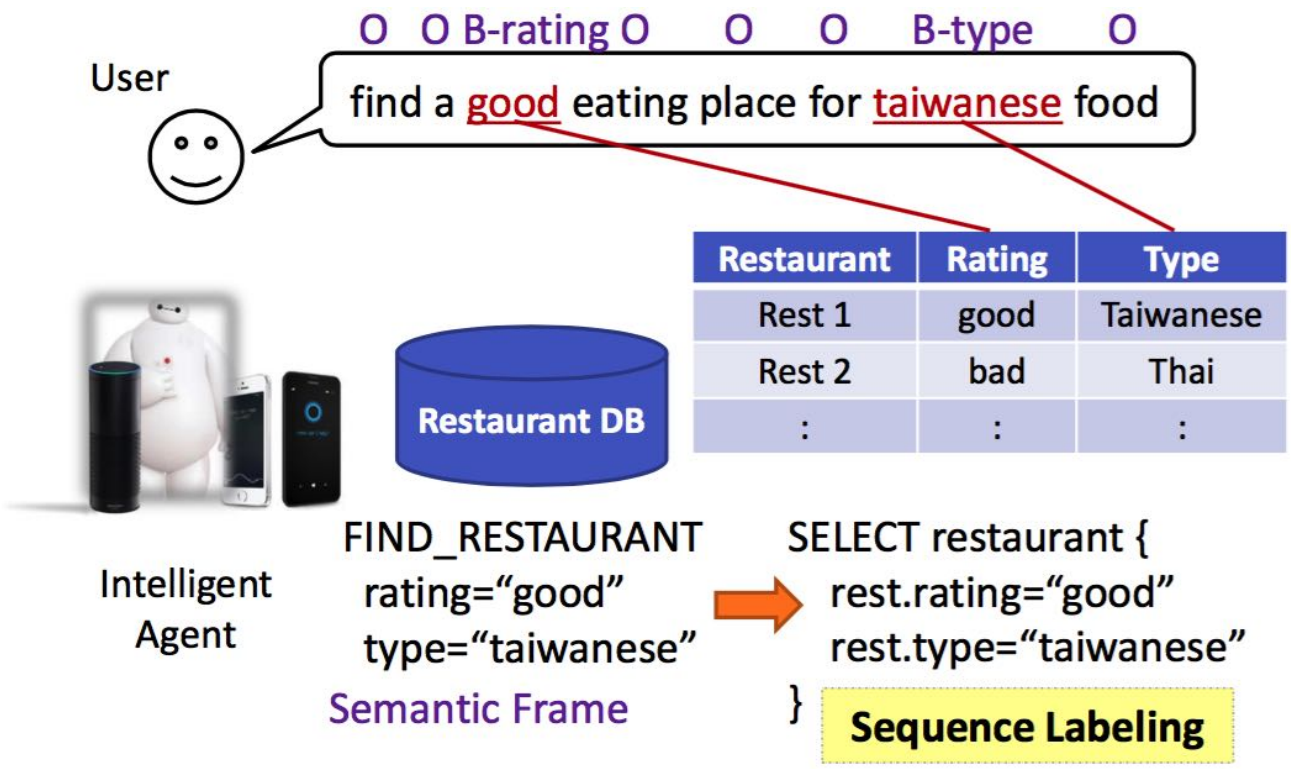


Dialogue System

Intent Detection

Dialogue System

Slot Filling



Dialogue System

Slot Filling

As a sequence tagging task

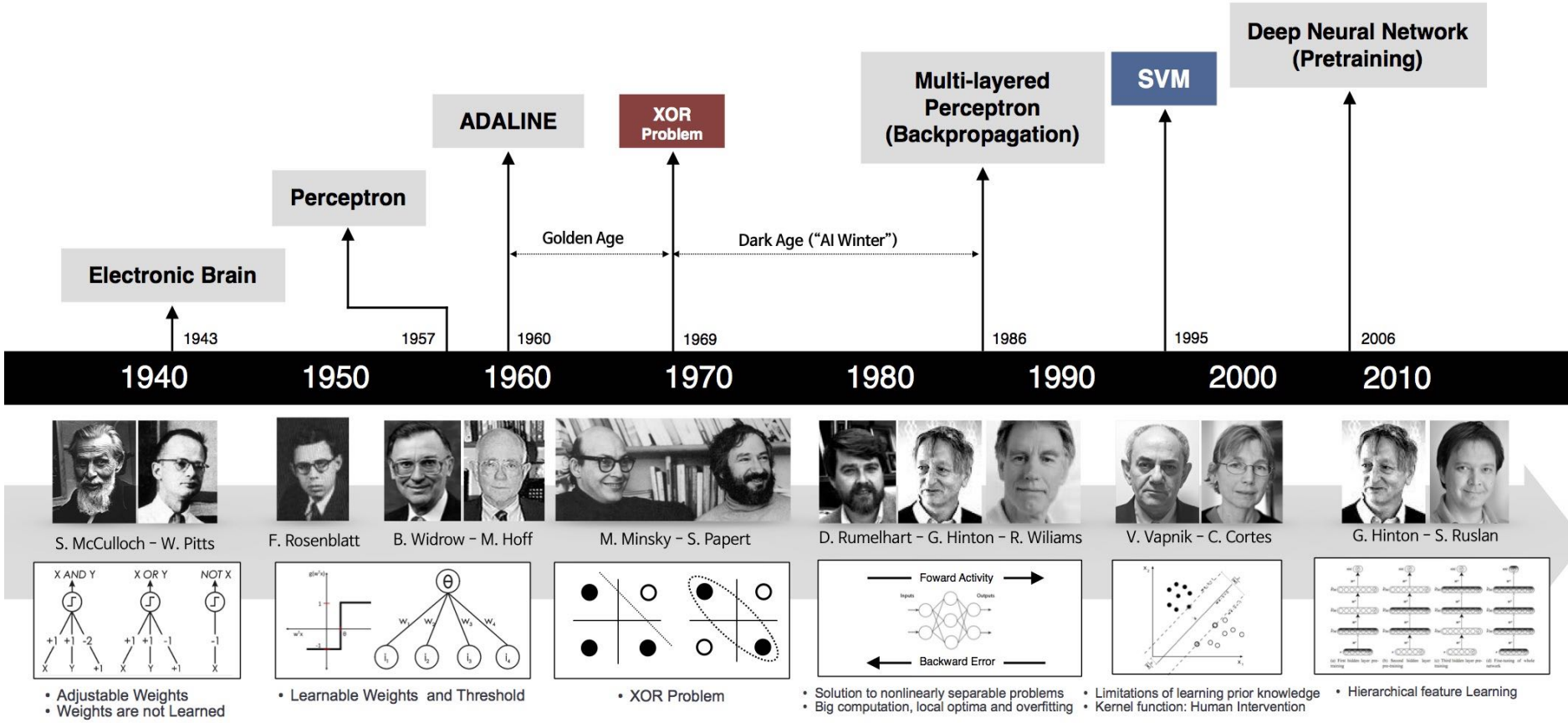
- Given a collection tagged word sequences, $S=\{((w_{1,1},w_{1,2},\ldots, w_{1,n1}), (t_{1,1},t_{1,2},\ldots,t_{1,n1})), ((w_{2,1},w_{2,2},\ldots,w_{2,n2}), (t_{2,1},t_{2,2},\ldots,t_{2,n2})) \ldots\}$ where $t_i \in M$, the goal is to estimate tags for a new word sequence.

flights from Boston to New York today

	flights	from	Boston	to	New	York	today
Entity Tag	O	O	B-city	O	B-city	I-city	O
Slot Tag	O	O	B-dept	O	B-arrival	I-arrival	B-date

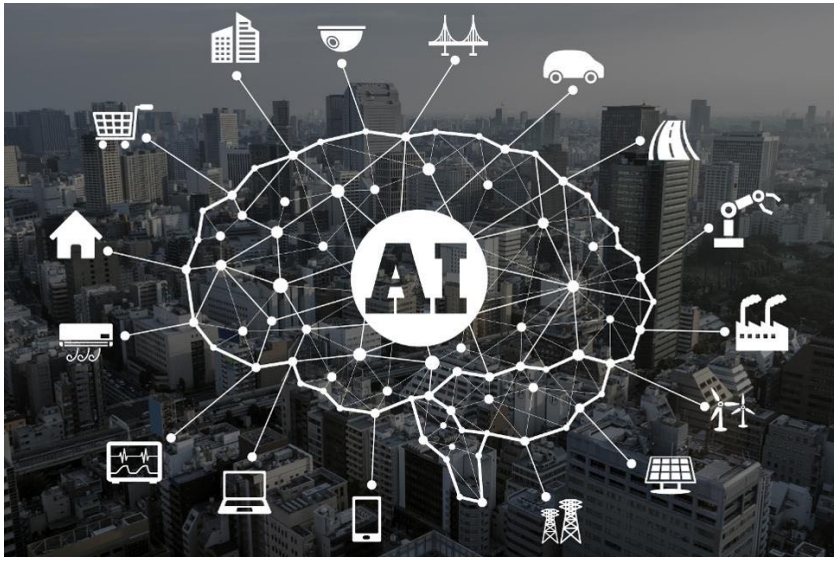
Introduction to Artificial Neural Network

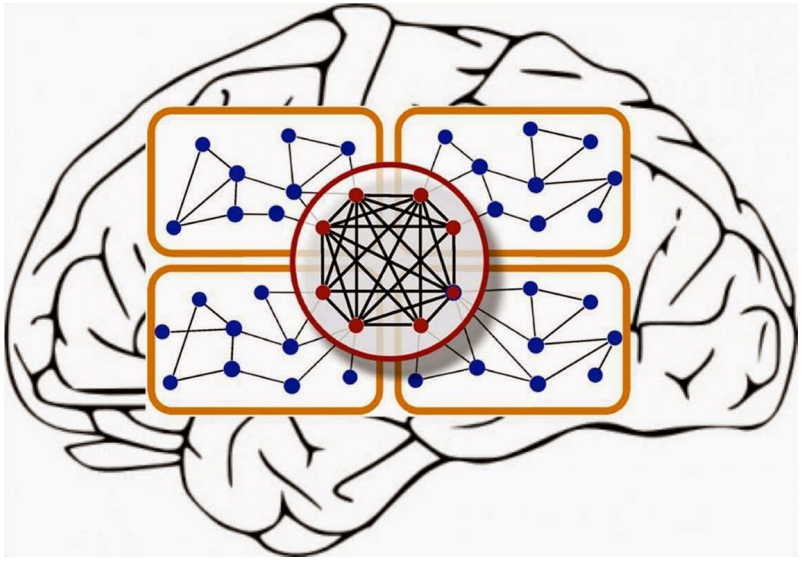
Milestones in the Development of Neural Networks



Biological Motivation (1/2)

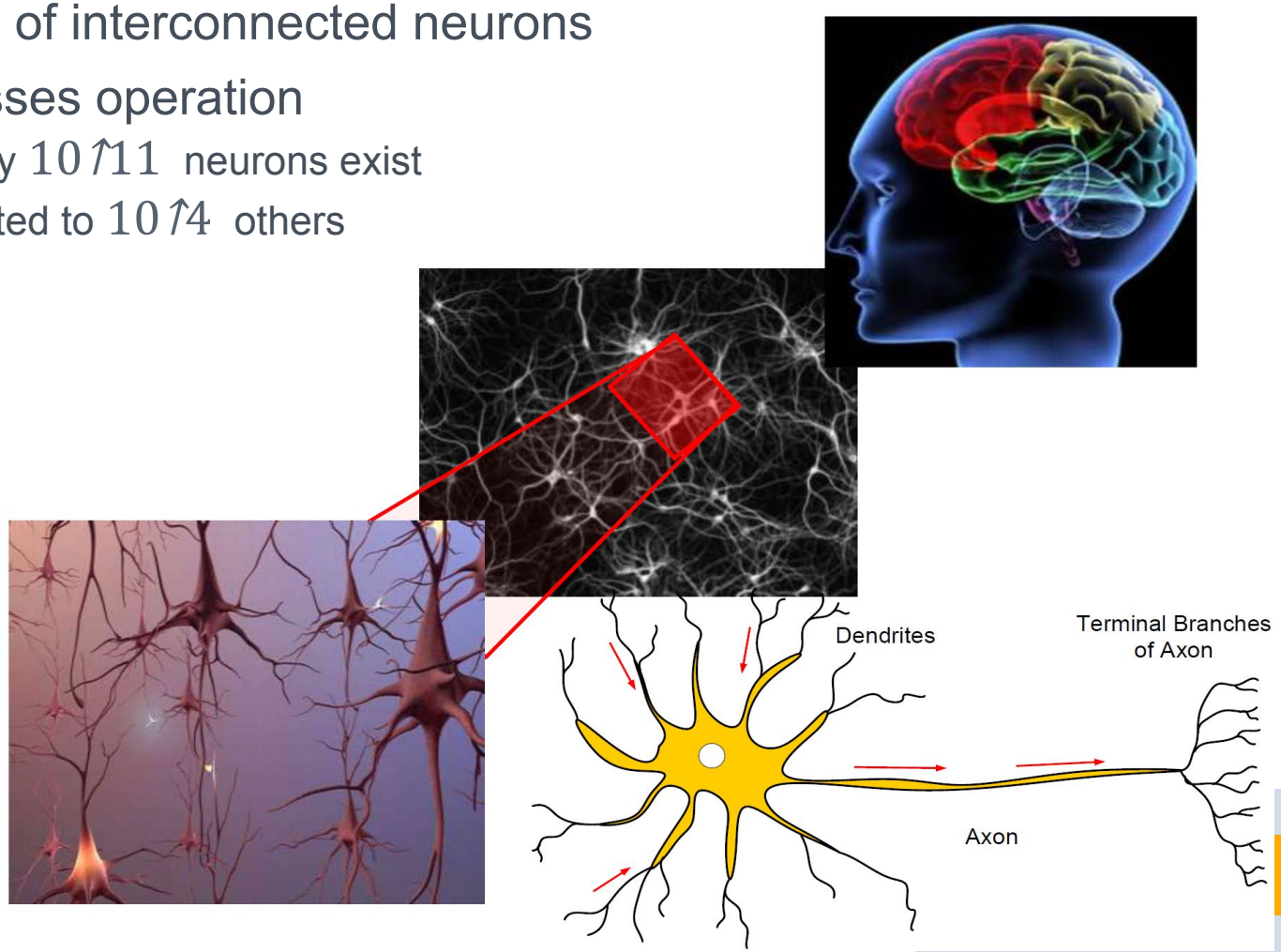
- Modeling of neural network in human brain





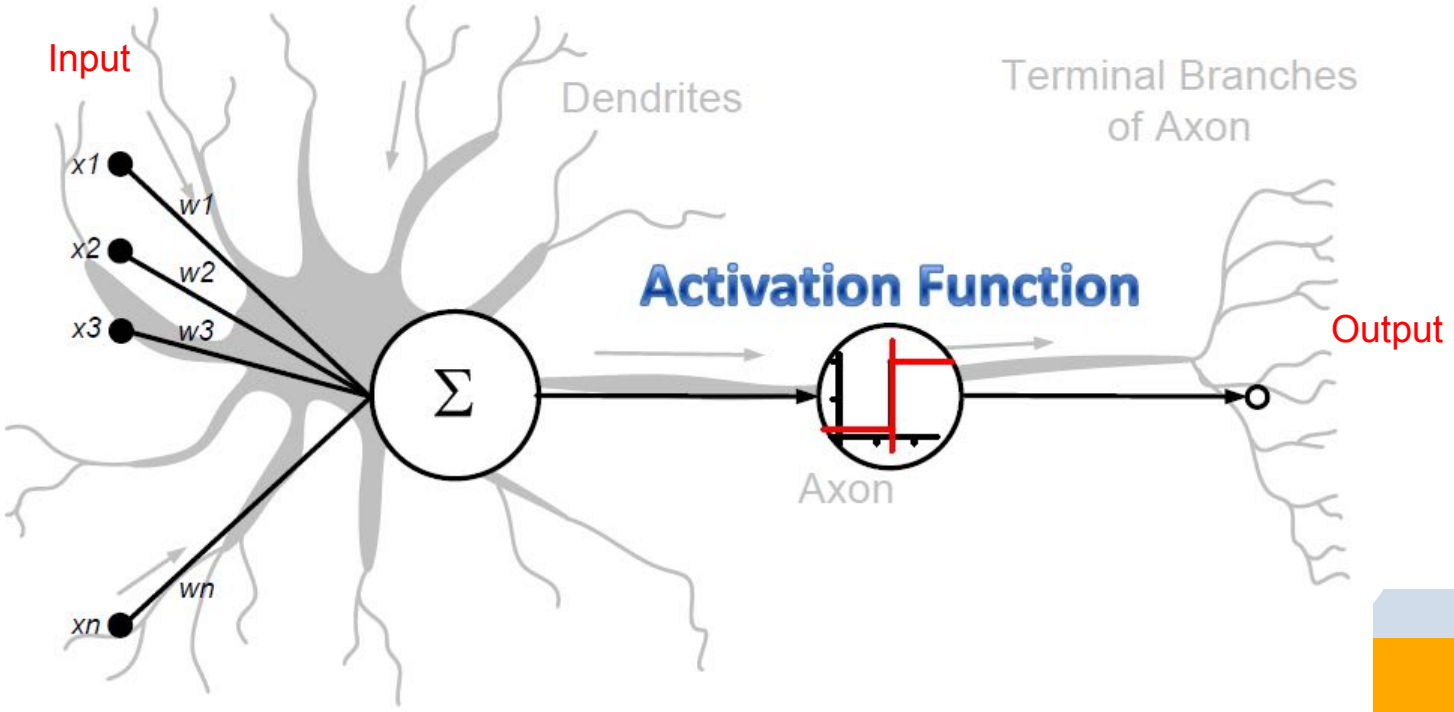
Biological Motivation (2/2)

- Complex webs of interconnected neurons
- Parallel processes operation
  - Approximately 10^{11} neurons exist
  - Each connected to 10^{4} others



Artificial neural network (ANN) (1/2)

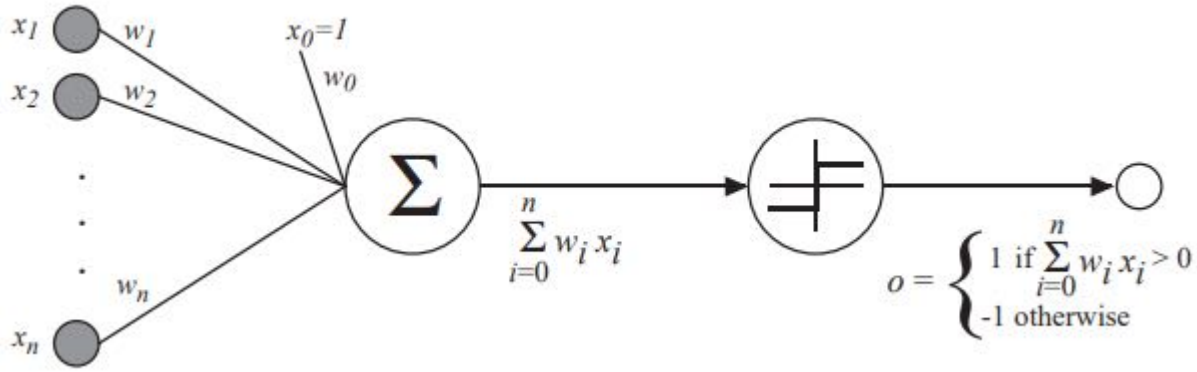
- Neurons integrate information
- Neurons pass information about the level of their input



Artificial neural network (ANN) (2/2)

- Brain structure is layered
- The influence of one neuron on another depends on the strength of the connection between them
- Learning is achieved by changing the strengths of connections between neurons

Perceptron



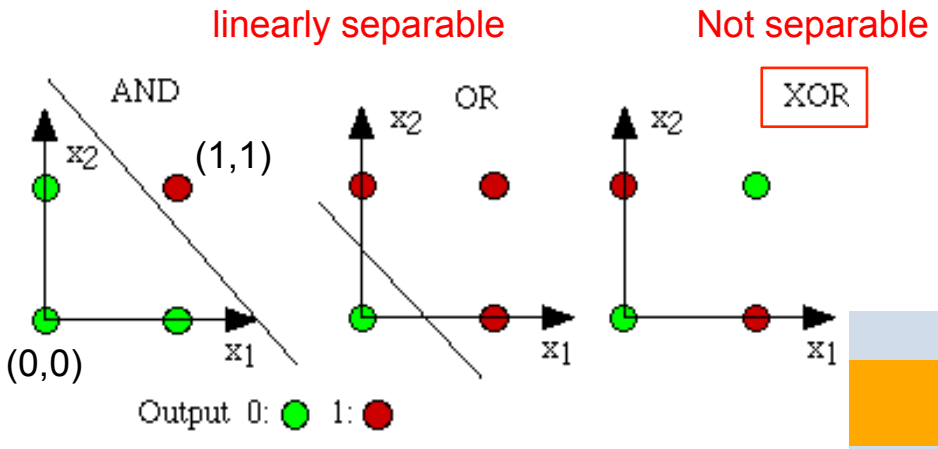
- Takes a vector of real-valued inputs ($x\downarrow 1$,...., $x\downarrow n$) weighted with ($w\downarrow 1$,...., $w\downarrow n$)
- Calculates the linear combination of these inputs
  - $\sum i=0 \uparrow n \blacksquare w\downarrow i \, x\downarrow i \; = w\downarrow 0 \; x\downarrow 0 + w\downarrow 1 \; x\downarrow 1 + ... + w\downarrow n \, x\downarrow n$
  - $w\downarrow 0$ denotes a threshold value (bias)
  - $x\downarrow 0$ is always 1
- Outputs 1 if the result is greater than 1, otherwise -1

Representation power of perceptron

- A perceptron represents a **hyperplane decision surface** in the n-dimensional space of instances
- Some sets of examples cannot be separated by any hyperplane, those that can be separated are called **linearly separable**
- Many Boolean functions can be represented by a perceptron: AND, OR, NAND, NOR

XOR

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0



The perceptron training rule

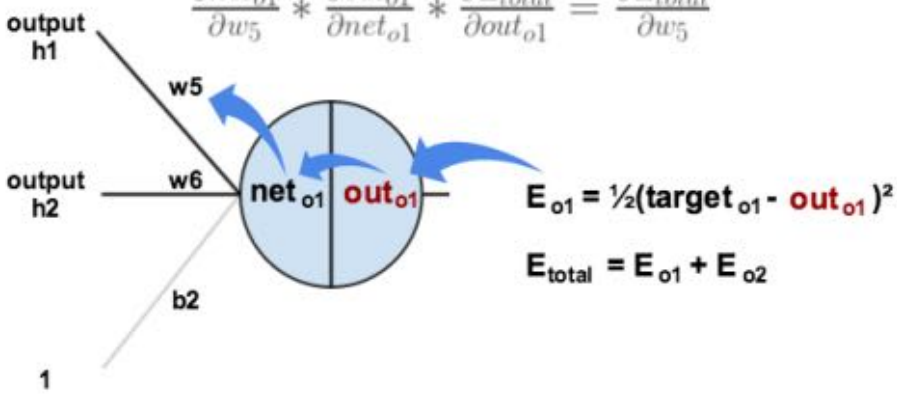
- **Problem** : determine a weight vector w that causes the perceptron to produce the correct output for each training example
- **Perceptron training rule :**
  - $w_i = w_i + \Delta w_i$ where $\Delta w_i = \eta(t-o)x_i$
  - t : target output, o : perceptron output
  - η : learning rate (usually some small value, e.g. 0.1)
- **Algorithm:**
  - Step 1. initialize w to random weights
  - Step 2. repeat, until each training example is classified correctly
- Convergence guaranteed provided **linearly separable training examples**

Delta rule (1/2)

- Perceptron rule fails if data is not linearly separable
- Delta rule for dealing with nonlinearity
  - Backpropagation using gradient descent algorithm

Chain rule

$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$

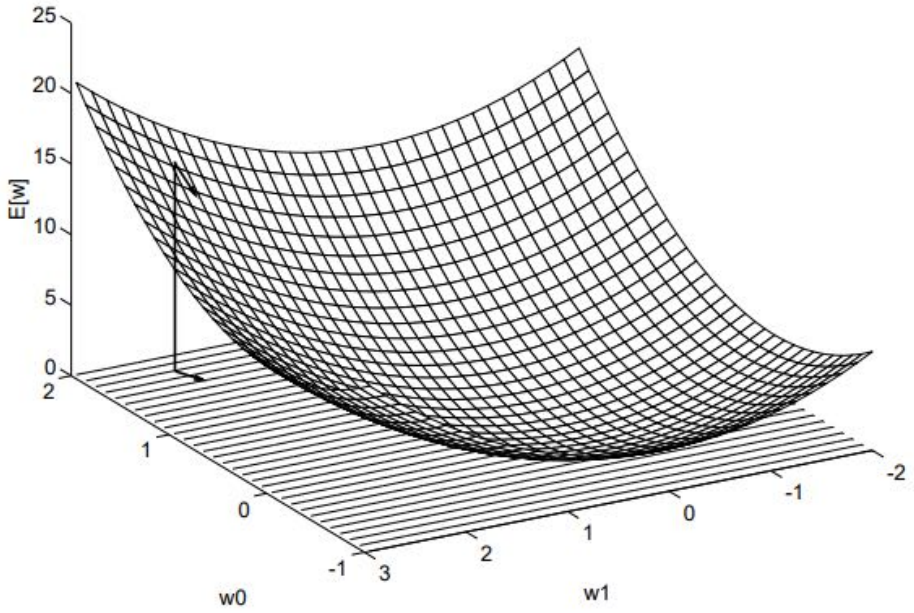


Delta rule (2/2)

- Use **gradient descent** to search the hypothesis space
  - Perceptron with step function as activation function cannot be used, because it is not differentiable
  - Hence, a **unthresholded linear unit** is appropriate
  - Error measure : $E(w) \equiv 1/2 \sum_{d \in D} (t_d - o_d)^2$
- To understand gradient descent, it is helpful to visualize the entire hypothesis space with
  - All possible weight vectors and
  - Associated E values

Error surface

- The axes $w\downarrow 0$, $w\downarrow 1$ represent possible values for the two weights of a simple linear unit



- An optimal error surface is **parabolic** (포물선) with a **single global minimum –** most real-world problems doesn't have an optimal error surface

Derivation of Gradient Descent (1/2)

- **problem** : How to calculate the **steepest descent** (내리막 경사) along the error surface? (error 값을 낮추는 쪽으로)
- Derivation of E with respect to each component of w
- This vector derivate is called **gradient of E**, written $\nabla E(w)$

$$\nabla E(\vec{w}) \equiv [\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots, \frac{\partial E}{\partial w_n}]$$

- $\nabla E(w)$ specifies the steepest ascent (오르막 경사), so $-\nabla E(w)$ specifies the steepest descent
- **Training rule** : $w_i = w_i + \Delta w_i$

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

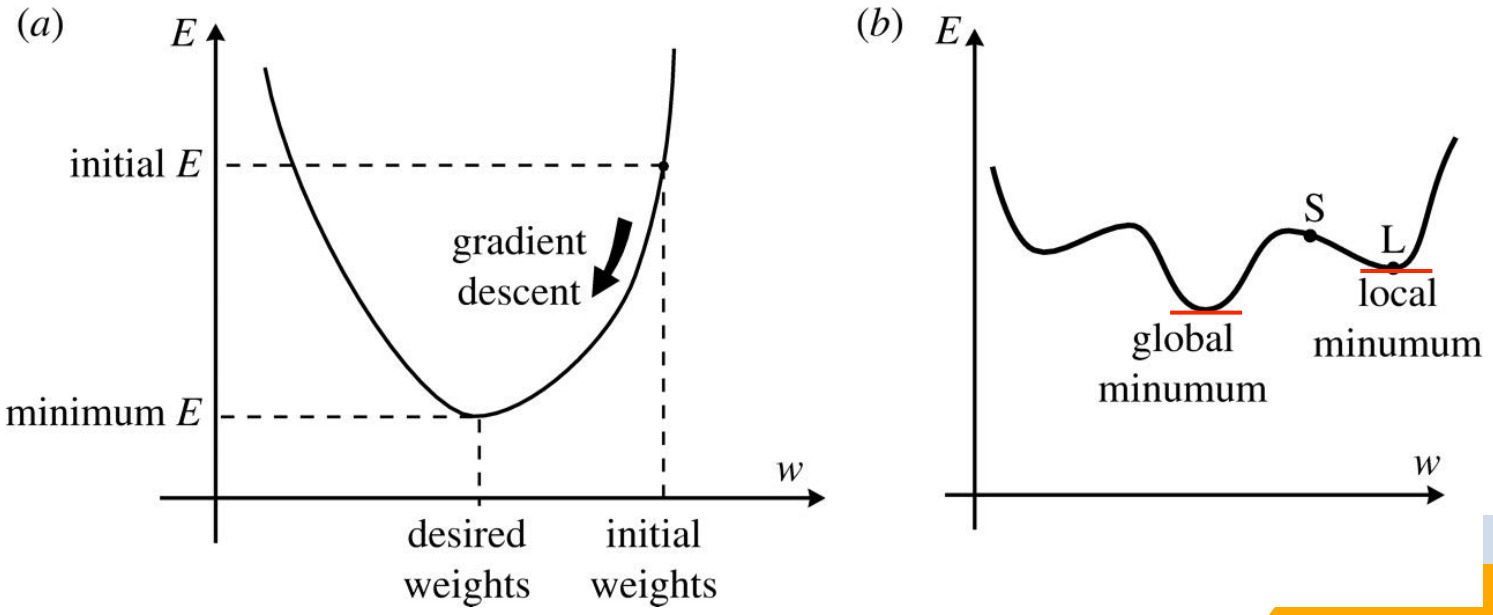
$$\Delta w_i \leftarrow \Delta w_i + \eta (t - o) x_i$$

Derivation of Gradient Descent (2/2)

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2}\sum_d (t_d - o_d)^2 \\
&= \frac{1}{2}\sum_d \frac{\partial}{\partial w_i}(t_d - o_d)^2 \\
&= \frac{1}{2}\sum_d 2(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - o_d) \\
&= \sum_d (t_d - o_d)\frac{\partial}{\partial w_i}(t_d - \vec{w}\cdot\vec{x_d}) \\
\frac{\partial E}{\partial w_i} &= \sum_d (t_d - o_d)(-x_{i,d})
\end{aligned}
$$

Difficulties in applying gradient descent

- If learning rate is large, overstepping problem
- If learning rate is small, local minima problem may occur



Stochastic Gradient Descent : SDG

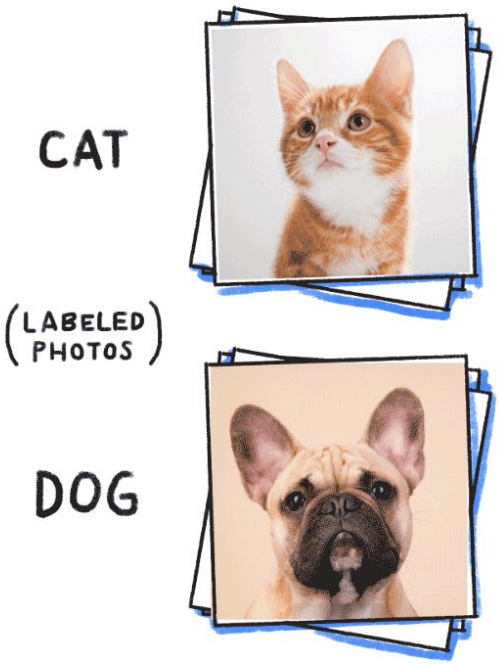
- **Idea** : updating weights right after applying an example (stochastic gradient descent)

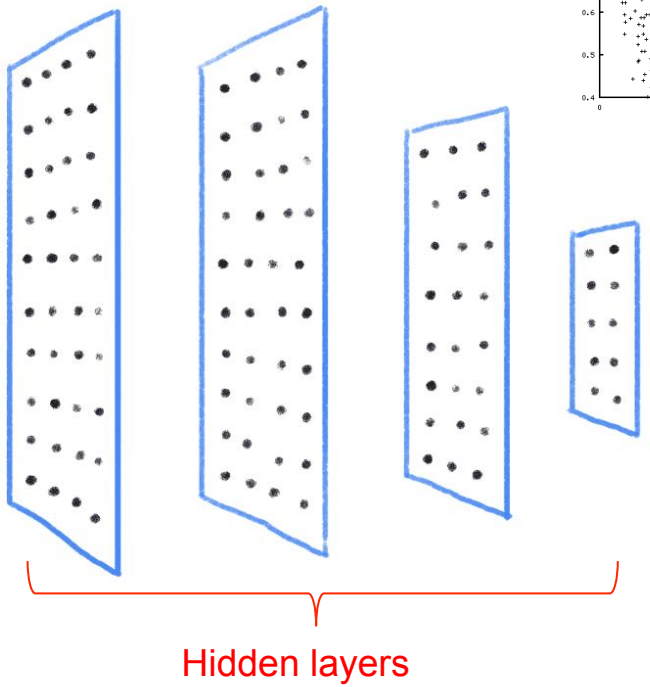
$$\Delta w_i = \eta(t-o)x_i \text{ where } E_d(\vec{w}) = \frac{1}{2}(t_d - o_d)^2$$

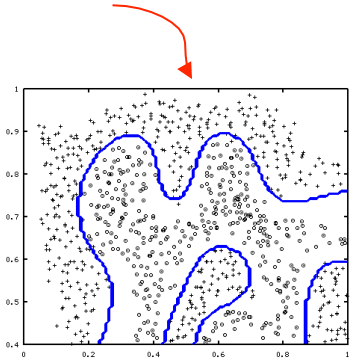
- Use smaller learning rate
- **Key differences:**
  - Gradient descent takes long time since weight updating is done only after applying every training example
    - SGD selects some training examples of which size is batch size, and update weights
  - Stochastic gradient descent requires less computation
  - Less probable of Local minima

Multilayer perceptron (MLP)

- Capable of learning nonlinear decision surfaces
  - Deep neural network

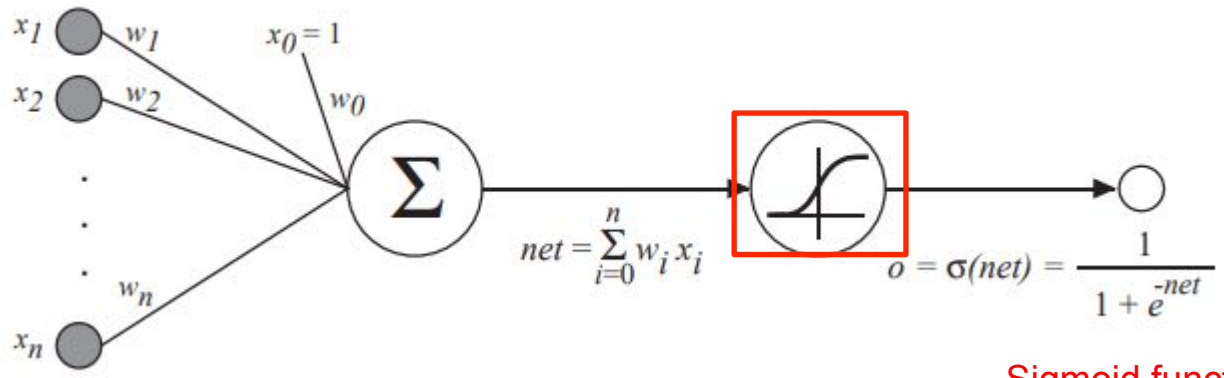






A differentiable threshold unit

- 비선형의 데이터에서 differentiable threshold unit을 사용함으로써 데이터를 구분 가능하도록 만듦
  - Activation function : sigmoid, ReLU ,….



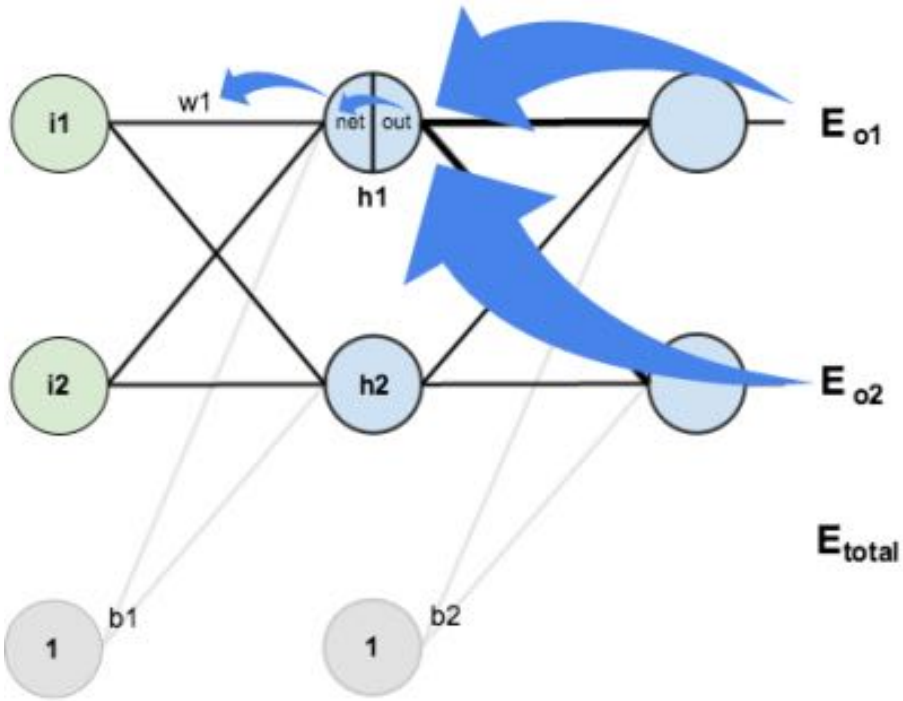
Sigmoid function

Backpropagation in MLP

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\downarrow$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$$E_{total} = E_{o1} + E_{o2}$$

Error gradient for a sigmoid unit

$$\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\
&= \frac{1}{2} \sum_d \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\
&= \frac{1}{2} \sum_d 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\
&= \sum_d (t_d - o_d) \left(-\frac{\partial o_d}{\partial w_i} \right) \\
&= -\sum_d (t_d - o_d) \frac{\partial o_d}{\partial net_d} \frac{\partial net_d}{\partial w_i}
\end{aligned}$$

Chain rule

Differentiation of sigmoid

$$\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x))$$

But we know:

$$\frac{\partial o_d}{\partial net_d} = \frac{\partial \sigma(net_d)}{\partial net_d} = o_d(1 - o_d)$$

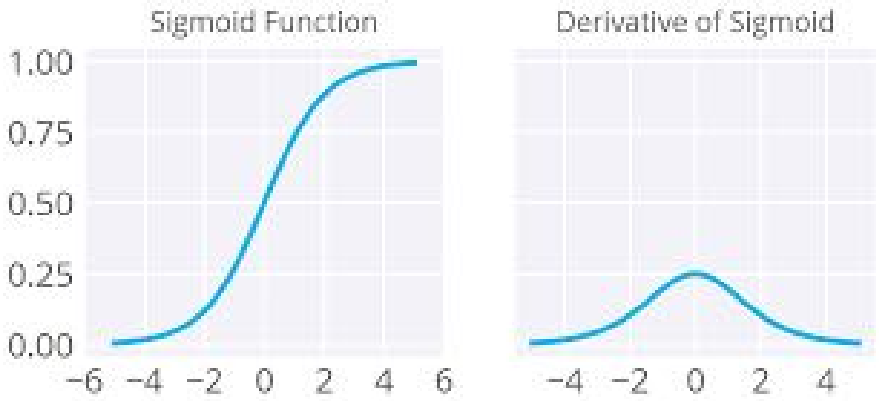
$$\frac{\partial net_d}{\partial w_i} = \frac{\partial (\vec{w} \cdot \vec{x}_d)}{\partial w_i} = x_{i,d}$$

So:

$$\frac{\partial E}{\partial w_i} = -\sum_{d \in D} (t_d - o_d) o_d (1 - o_d) x_{i,d}$$

Activation functions

- Sigmoid function is basic activation function
  - Sigmoid function : $f(x)=1/1+e\uparrow -x$, $df/dx=e\uparrow x/(e\uparrow x+1)\uparrow 2 \Rightarrow 0<df/dx\leq 0.25$
  - When network is deep, vanishing gradient problem occurs



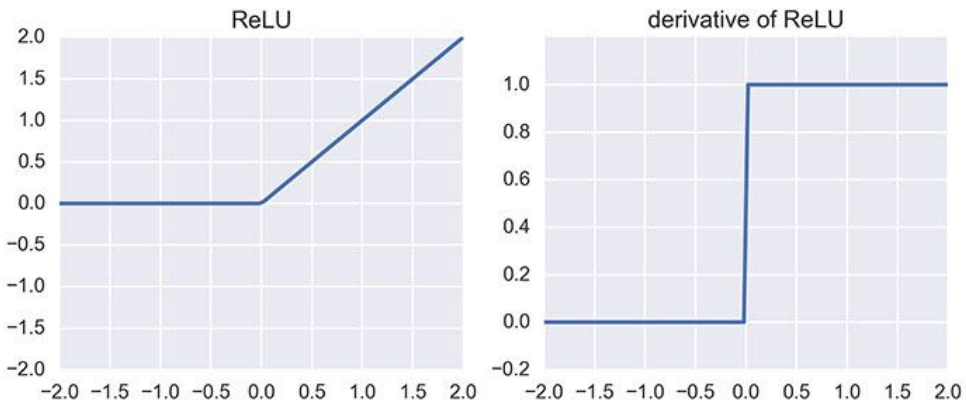
Vanishing gradient problem

- Error를 계산할 때, Chain rule을 사용하여 gradient를 계산하는 과정에서 0과 1 사이의 수를 반복하여 곱하게 됨
  - Deep structure의 경우 저층 layer의 gradient가 너무 작아져서 parameter update가 매우 느려짐

- 해결책
  - Vanishing/exploding gradient effect를 최소화할 수 있는 값으로 각 layer를 초기화 (e.g., Xavier initialization (Glorot et al., 2010))
  - 더 빠른 하드웨어를 사용하여 NN을 많은 epoch동안 훈련
  - **Derivate $f(x)$ ($i.e.$, $f'(x)$) 가 1인 구간이 있는 activation function 사용 (대표적으로 ReLU function) => 아예 1을 만들어 버리자**

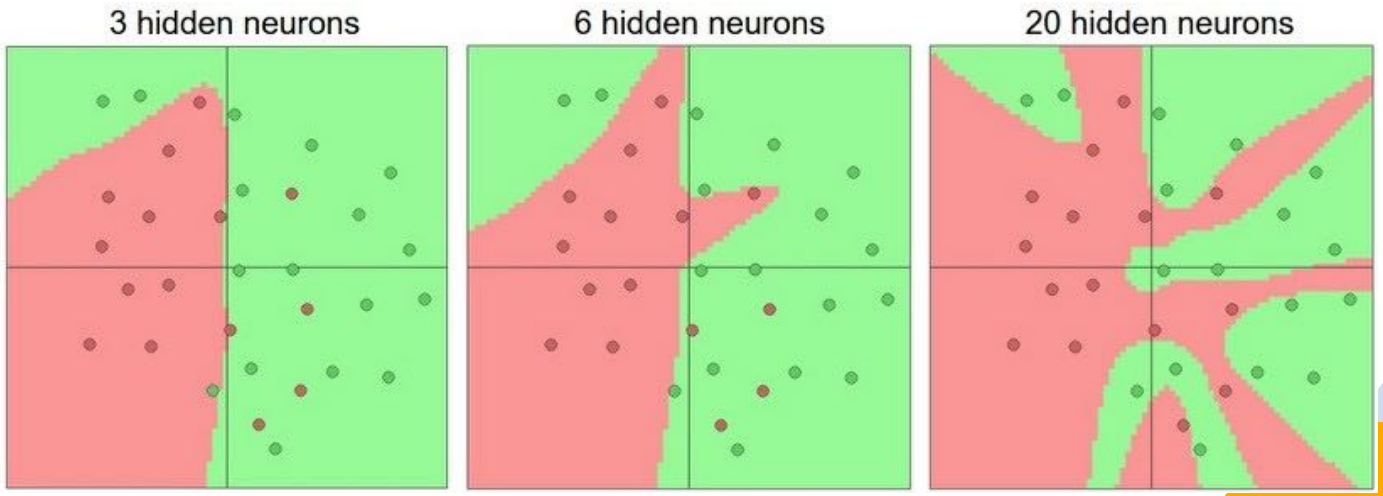
Rectified Linear Units

- ReLU function : $f(x)=\max(0, x)$, $df/dx = \begin{cases} 1 & if\ x>0 \\ 0 & if\ x\leq 0 \end{cases}$
- 0보다 작을 때는 0을 사용하고, 0보다 큰 값에 대해서는 해당 값 그대로 사용
- $x\geq 0$인 경우 $df/dx = 1$ 이므로 deep NN의 저층 layer의 gradient를 계산할 때 gradient가 소실되거나 증폭되는 문제가 완화됨



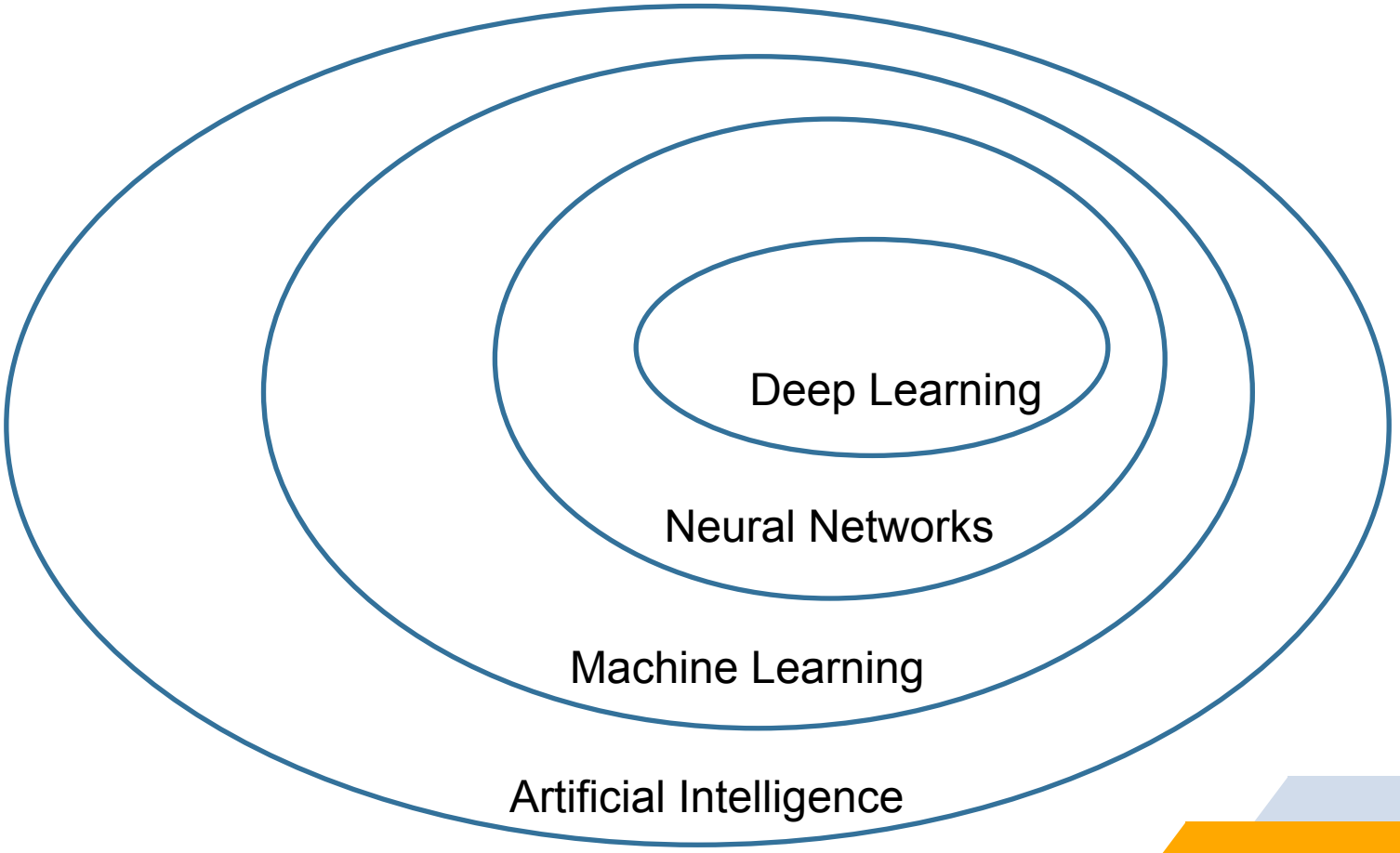
Number of hidden units and layers

- Hidden layer과 neuron의 개수는 몇 개를 해야할까?
- Neural network에서 layer의 수와 크기를 증가시킴으로써 network의 수용량은 커짐
- 장점 : 더 복잡한 데이터를 분류하기 위해 학습할 수 있음
- 단점 : 과도한 학습으로 인해 쉽게 overfitting될 수 있음



Popular and Representative Deep Learning Models

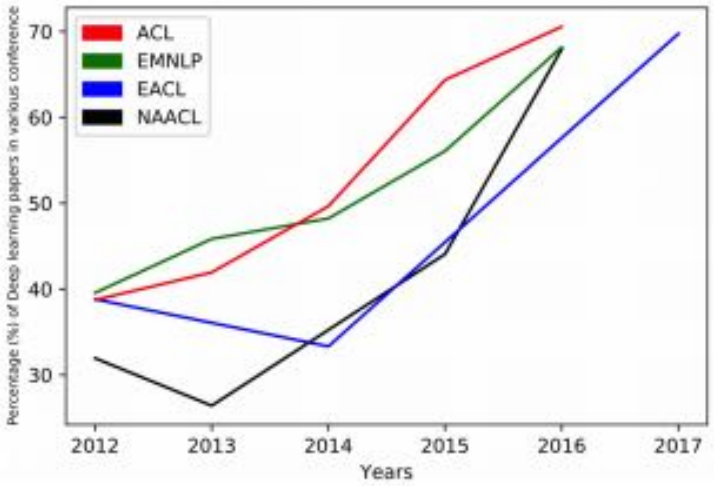
Topology of Deep Learning



Trends in Deep Learning Based NLP

Deep Learning and NLP

- Recent NLP research is increasingly focusing on the use of new deep learning methods.
  - NLP problems have been based on shallow models trained on very high dimensional and sparse features.
  - But recently neural networks have been producing superior results on NLP tasks.
- Percentage of deep learning papers in ACL, EMNLP, EACL, NAACL over the last 6 years have been rapidly increasing.



Trends in Deep Learning Based NLP

Deep Learning based NLP

- Distributed Representation
  - Word Embeddings, Word2Vec, Character Embeddings
- Convolutional Neural Network
  - Basic CNN network, Applications
- Recurrent Neural Network
  - Need for Recurrent Networks, RNN models, Applications, Attention Mechanism
- Recursive Neural Networks
- Deep Reinforced Models and Deep Unsupervised Learning
  - Reinforcement Learning for sequence generation, Unsupervised sentence representation learning, Deep Generative models
- Memory Augmented Networks
- Line ups
  - POS tagging, Parsing, Named-Entity Recognition, Semantic Role Labeling, Sentiment Classification, Machine Translation, Question Answering, Dialog Systems
- Future Trends

NLP Trends considering CNN

Convolutional Neural Networks

- As the need arose for an effective feature function that extracts higher-level features from constituting words or n-grams, CNN turned out to be the best choice.
  - Given the CNN's effectiveness in computer vision task, it had been seen suitable for numerous NLP tasks such as sentiment analysis, summarization, machine translation, and question answering.
- The use of CNNs for sentence modeling traces back to (Collobert and Weston, 2008).
  - Used multi-task learning to output multiple predictions for NLP tasks such as POS tags, chunks, named-entity tags, semantic roles, semantically similar words and a language model.
- CNNs have the ability to extract salient n-gram features from the input sentence to create an informative latent semantic representation of the sentence for downstream tasks.

NLP Trends considering CNN

CNN NLP Application

- The utilization of CNN in sentence modeling was done by Kim, 2014 for a variety of sentence classification tasks with competitive results.
  - Yet it still had many shortcomings with the CNN's inability to model long distance dependencies standing as the main issue.
- Kalchbrenner et al., 2014 proposed a Dynamic CNN(DCNN) for semantic modeling of sentences.
  - Primarily proposed dynamic k-max pooling strategy which, given a sequence p selects the k most active features.
  - The selection preserved the order of the features but was insensitive to their specific position.

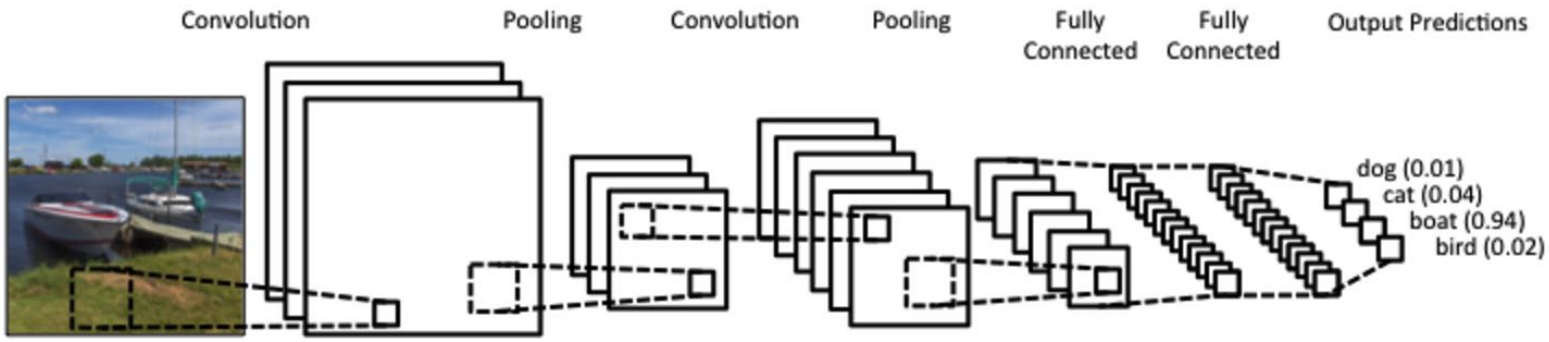
NLP Trends considering CNN

CNN NLP Application

- CNN models are also suitable for certain NLP tasks that require semantic matching beyond classification.
- This is because…
  - CNNs are wired in a way to capture the most important information in a sentence.
    → However, this often misses valuable information present in multiple facts within the sentence.
  - CNNs inherently provide certain required features like local connectivity, weight sharing, and pooling.
    → This puts forward some degree of invariance which is highly desired in many tasks.

- Overall, CNNs are extremely effective in mining semantic clues in contextual windows.
  - However, as they are very data heavy, they are also pose a problem when scarcity of data arises.

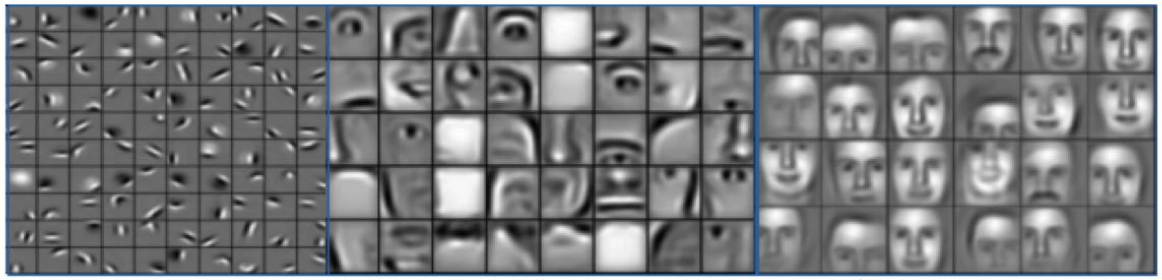
Convolutional Neural Network(CNN)

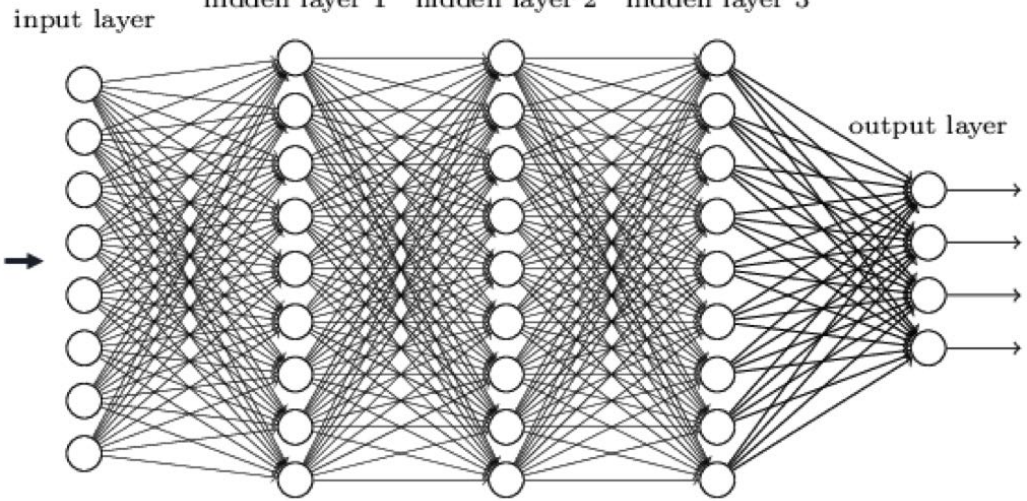


- Extract features through convolution and pooling
- Generally hundreds to thousands of filters (kernels) are used
- It has mainly been applied to image processing, but it is also applied to NLP task recently.

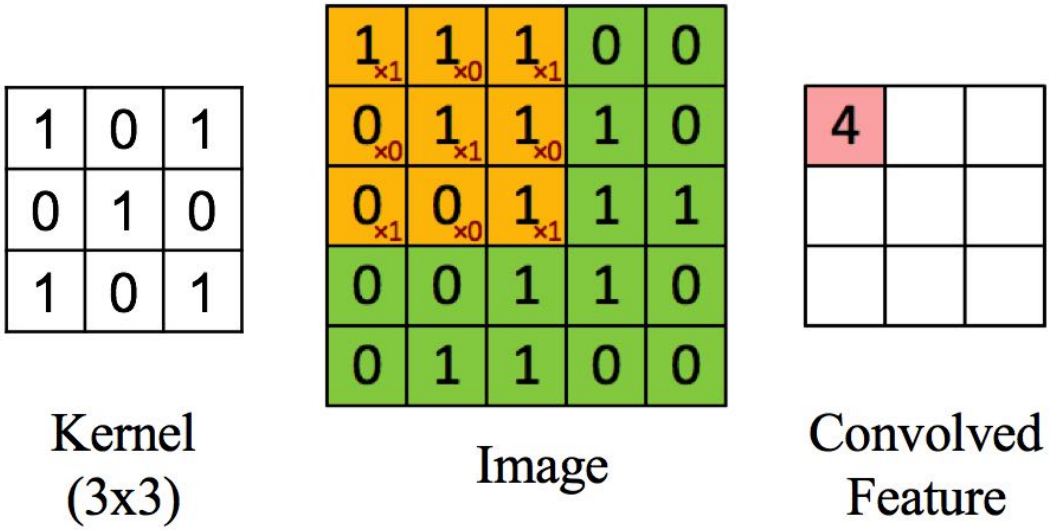
Convolutional Neural Network(CNN)

Deep neural networks learn hierarchical feature representations



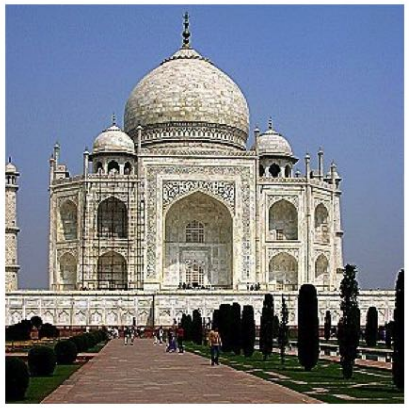


Convolution



- Mainly used for image processing
- Filter matrix value multiplied by the pixel value of the image.

Convolution



0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

Sharpen

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

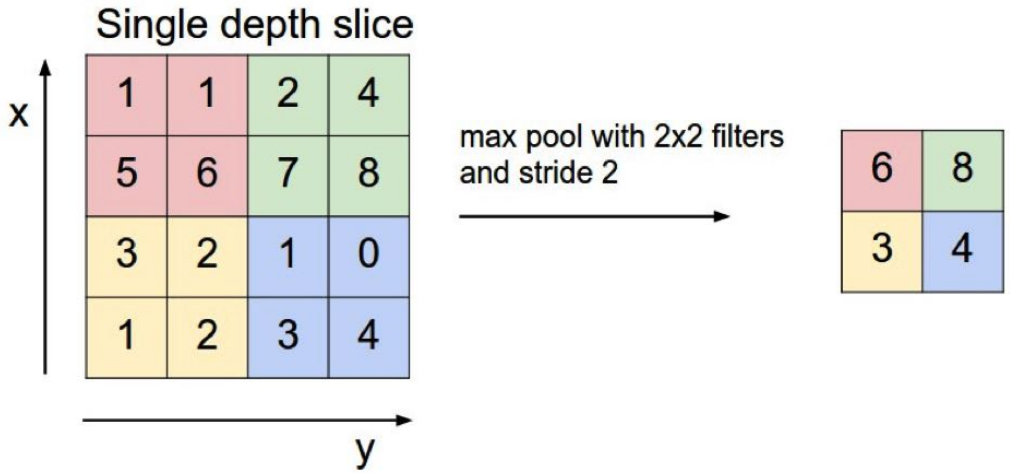
Blur

	0	1	0	
	1	-4	1	
	0	1	0	

Edge detect

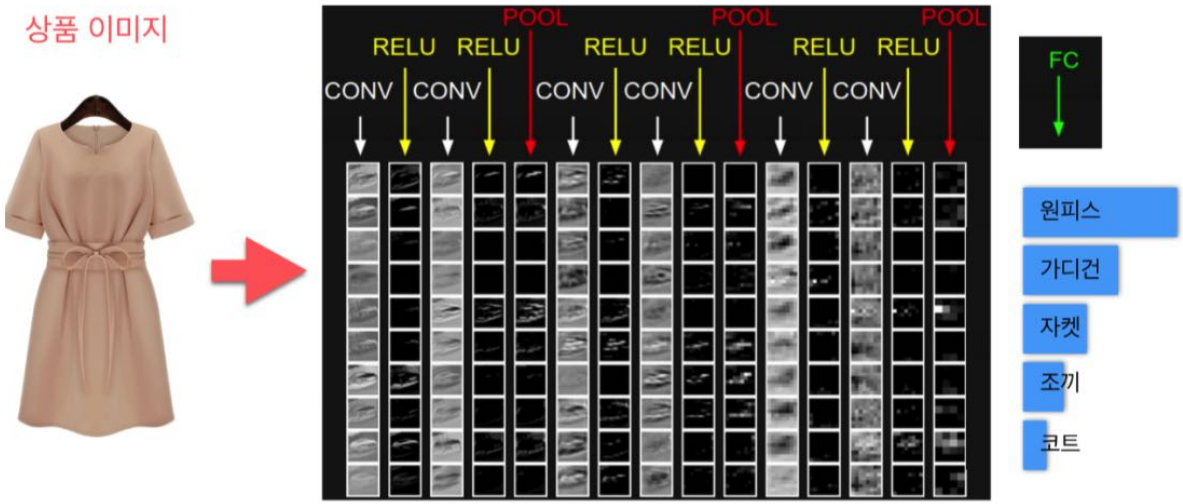
- Depending on the value of the filter, the image can have various effects.

Pooling



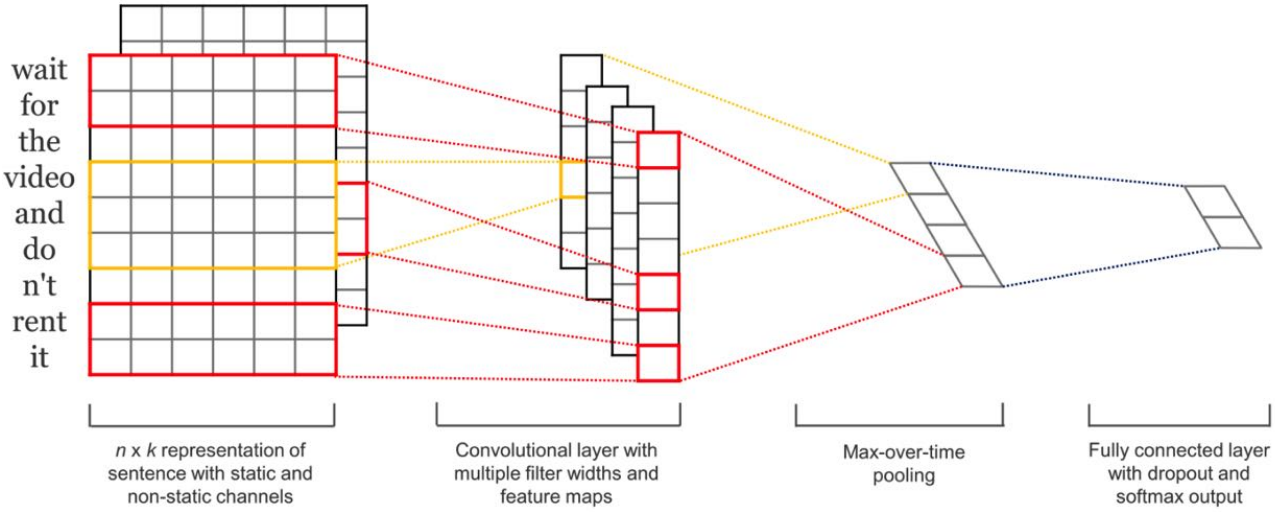
- Reduce the size of the feature
- The most commonly used max pooling preserves only the maximum value of the values in the filter.

Image Classification using CNN



- Can be used for image classification by using final qualities extracted through Convolution and pooling process

CNN for NLP



- Concatenate the vectors of the words constituting the sentence, and apply CNN
- Unlike RNN, only the number of words in the same size as the filter size can be reflected

NLP Trends considering RNN

Recurrent Neural Networks

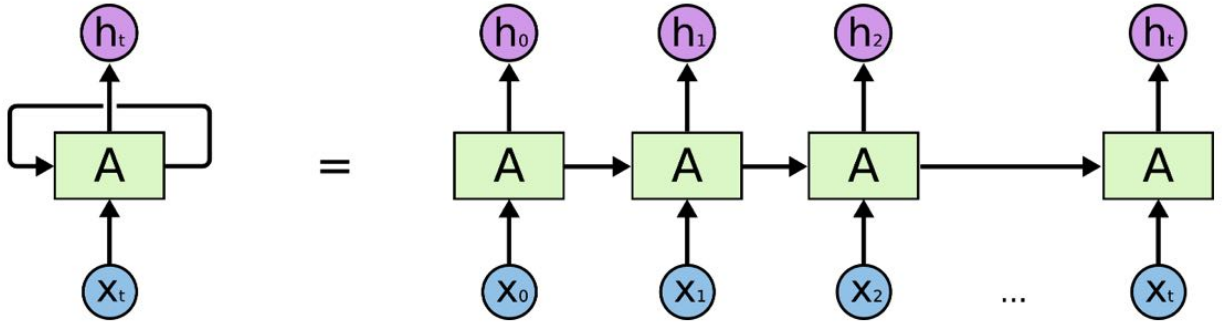
- RNNs work around the idea of processing sequential information.
- Need for Recurrent Networks
  - Given that a RNN performs sequential processing by modeling units in sequence, it has the ability to capture the inherent sequential nature present in language, where units are characters, words or even sentences.
  - In a way, RNNs have “memory” over previous computations and use this information in current processing.
    → And words in a language develop their semantical meaning based on the previous words in the sentence.
  - It also has the ability to model variable length of text, including very long sentences, paragraphs and even documents.
  - It is also apt for creating a gist of the sentence in a fixed dimensional hyperspace.
    → Essential as Many NLP tasks also requires semantic modeling over the whole sentence.
- RNNs try to create a composition of an arbitrarily long sentence along with unbounded context.

NLP Trends considering RNN

RNN NLP Applications

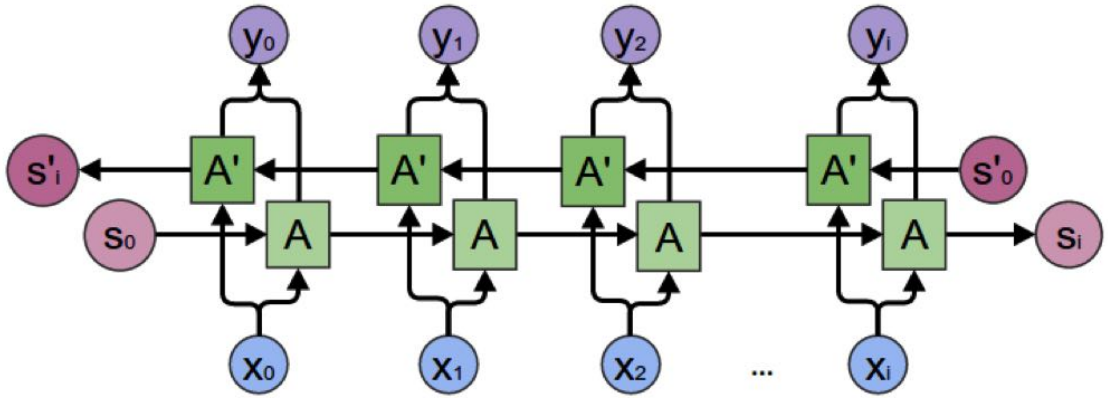
- RNN for word-level classification:
  - Has a huge presence in the field of word-level classification.
    → Lample et al., 2016 proposed to use bidirectional LSTM for NER
  - Have also shown considerable improvement in language modeling over traditional methods based on count statistics.
    → Graves,2013 introduced the effectiveness of RNNs in modeling complex sequences with long range context structures.
- RNN for sentence-level classification:
  - Studies show that the dynamics of LSTM gates can capture the reversal effect of the word ‘not’.
  - The hidden state of a RNN can be used for semantic matching between texts.
- RNN for language generation:
  - Conditioned on textual or visual data, deep LSTMs have been shown to generated reasonable task specific text in tasks such as machine translation, image captioning and etc.

Recurrent Neural Network (RNN)



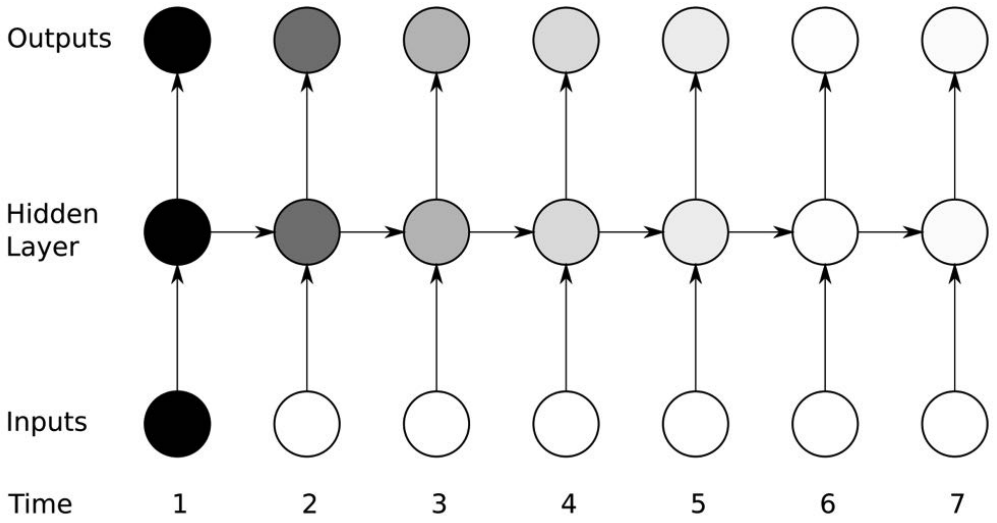
- A kind of neural network with cyclic structure
- A circular structure results in a state, which can handle sequences of varying lengths.
- Output $h\downarrow t$, reflecting the input value $[x\downarrow 0$, $x\downarrow 1$, ...,$x\downarrow t-1$ $]$

Bidirectional RNN



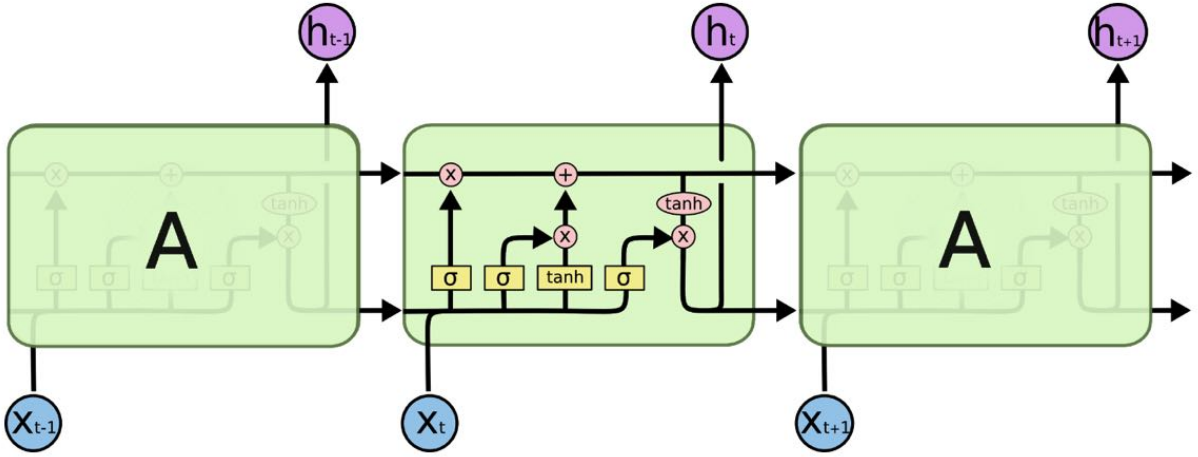
- RNNs that combine two RNNs in different directions to enable bidirectional dependence
- Output $y\downarrow t$ has input $[x\downarrow 0 , x\downarrow 1 , ..., x\downarrow t-1]$ and $[x\downarrow t+1 , x\downarrow t+2 , ..., x\downarrow N]$ is reflected

Long-term Dependency Problem



- The basic RNN is diluted with the state value as the step progresses, making it difficult to reflect long-distance dependency

Long Short-Term Memory (LSTM)



- An improved RNN structure to reflect long-distance dependency by adding a gate to control the amount of information transfer without reflecting the input value unconditionally to the state

Long Short-Term Memory (LSTM)

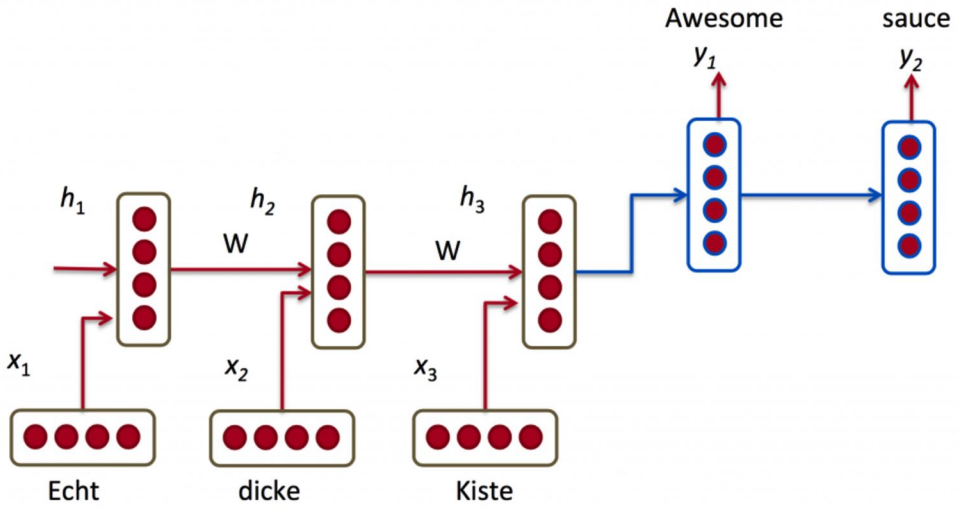
$$
(0,1)\begin{cases} i_t = \sigma(W_i[h_{t-1}, x_t] + b_i & \text{Input gate} \\ f_t = \sigma(W_f[h_{t-1}, x_t] + b_f & \text{Forget gate} \\ o_t = \sigma(W_o[h_{t-1}, x_t] + b_o & \text{Output gate} \end{cases}
$$

$$
l_t = \tanh(W_l[h_{t-1}, x_t] + b_l \quad \text{New input}
$$

$$
\tilde{h}_t = f_t \cdot \tilde{h}_{t-1} + i_t \cdot l_t \quad \text{Hidden state update}
$$

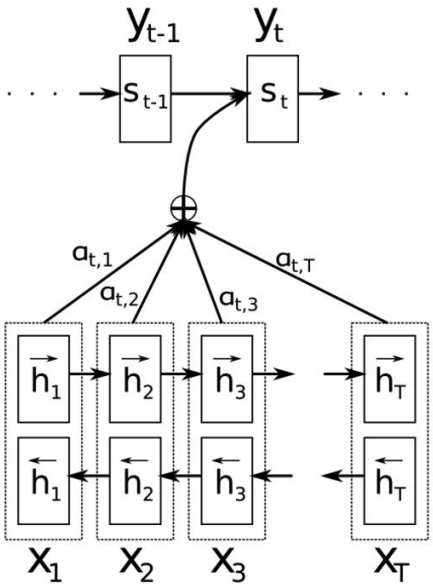
$$
h_t^s = o_t \cdot \tilde{h}_t \quad \text{New output}
$$

RNN without Attention



- For an RNN without an Attention structure, the decoder generates an output sentence based on the last hidden state ($h↓3$) of the encoder
- However, there is a limit to expressing the input sentence by only $h↓3$

RNN with Attention



$$p(y_i|y_1,\ldots,y_{i-1},\mathbf{x}) = g(y_{i-1},s_i,c_i)$$

$$s_i = f(s_{i-1},y_{i-1},c_i)$$

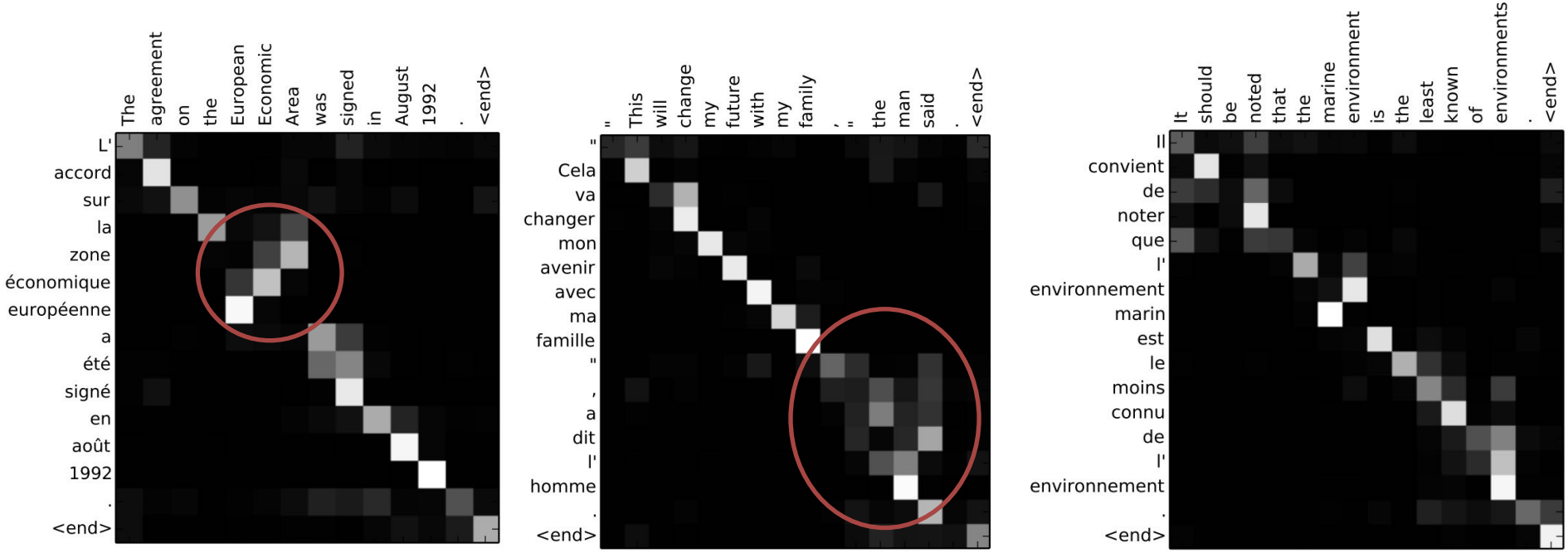
$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

$$e_{ij} = a(s_{i-1},h_j)$$

- Consider the weighted combination of all the input states, not just the last hidden state.

Visualization of Attention Weight Matrix in NMT

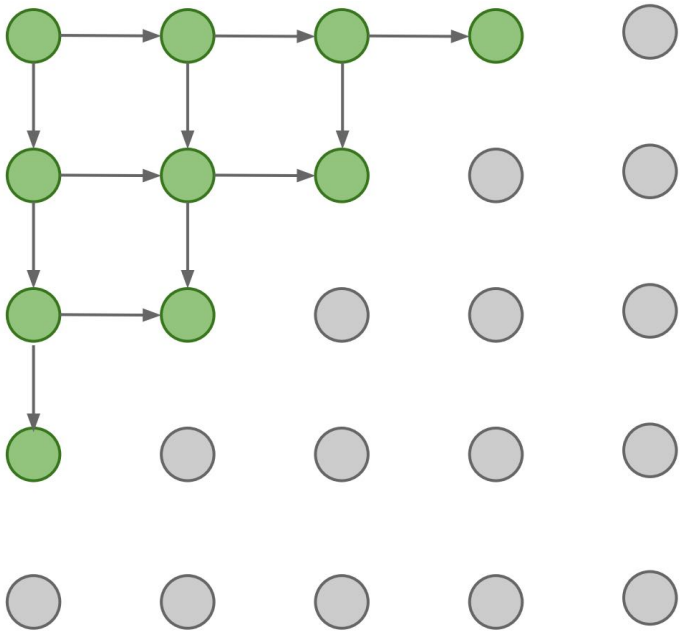


- Visualizing the attention weight in English-French translation. The model learns to reorder words and look at multiple words simultaneously.

PixelRNN

- Generate image pixels starting from corner
- Dependency on previous pixels modeled using an RNN (LSTM)
- Drawback: sequential generation is slow!

- Oord, A. V. D., Kalchbrenner, N., & Kavukcuoglu, K. (2016). Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*.

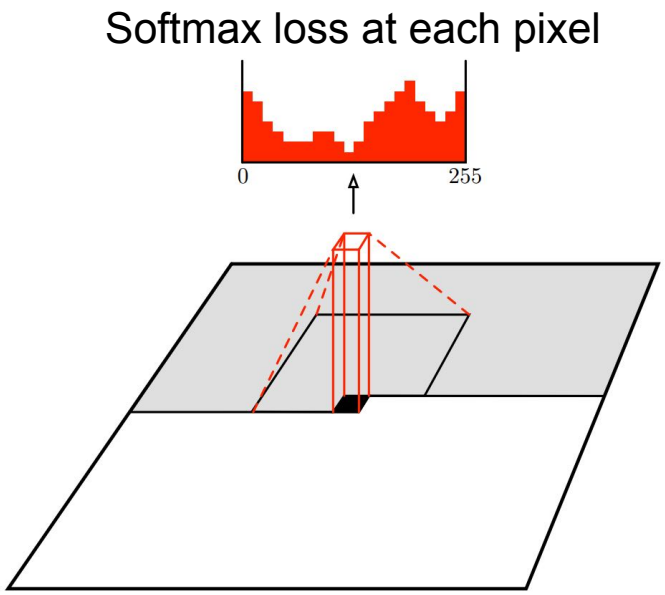


PixelRNN - Result

- Generating full images from partially occluded input

PixelCNN

- Similar to PixelRNN, but Dependency on previous pixels modeled using a CNN over context region
- Training is faster than PixelRNN
  - can parallelize convolutions since context region values known from training images
- Generation must proceed sequentially => still slow



- van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., & Graves, A. (2016). Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems* (pp. 4790-4798).

PixelCNN - Result

- Generating images conditioned on class label

Grey whale

Tiger

EntleBucher (dog)

Yellow lady's slipper (flower)

PixelRNN and PixelCNN

- Pros
  - Can explicitly compute likelihood p(x)
  - Explicit likelihood of training data gives good evaluation metric
  - Good samples

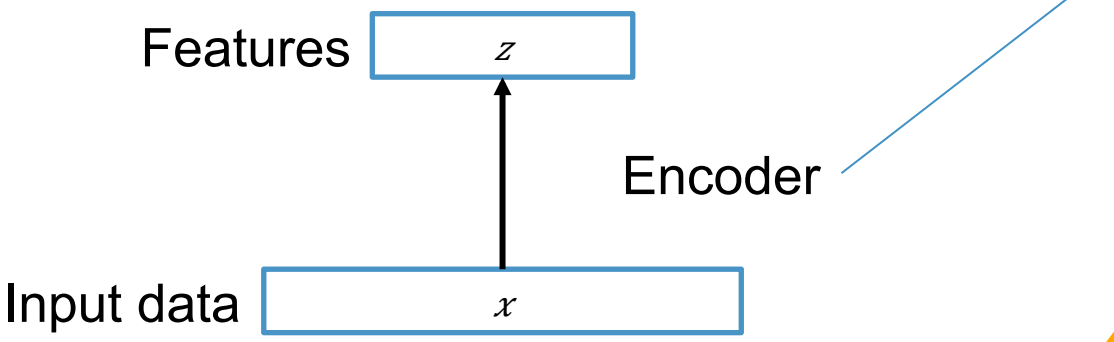
- Con
  - Sequential generation => slow

Autoencoders

- Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

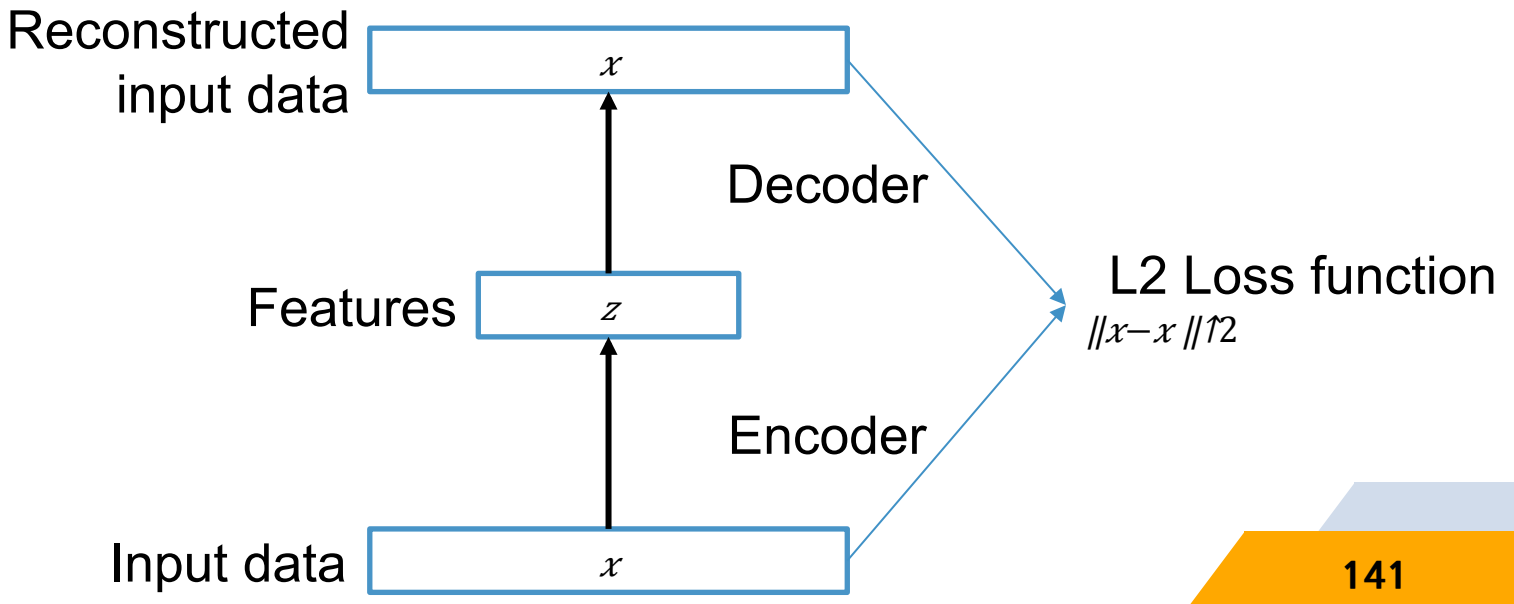
z is usually smaller than x (dimensionality reduction)
Q: Why dimensionality reduction?
A: Want features to capture meaningful factors of variation in data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN



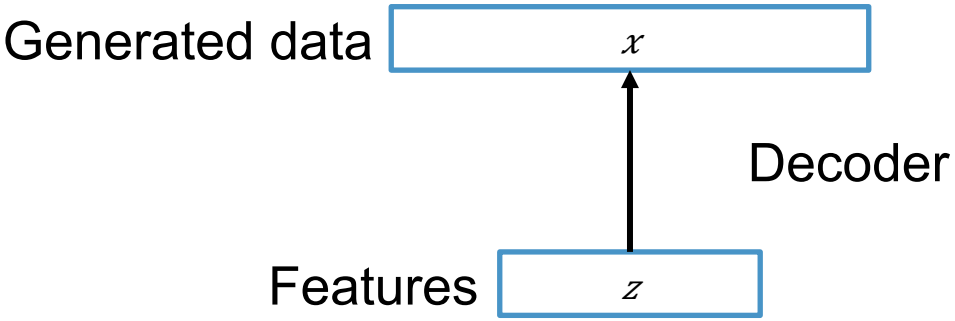
Autoencoders

- How to learn this feature representation? Train such that features can be used to reconstruct original data
  - "Autoencoding" - encoding itself (doesn't require labels)



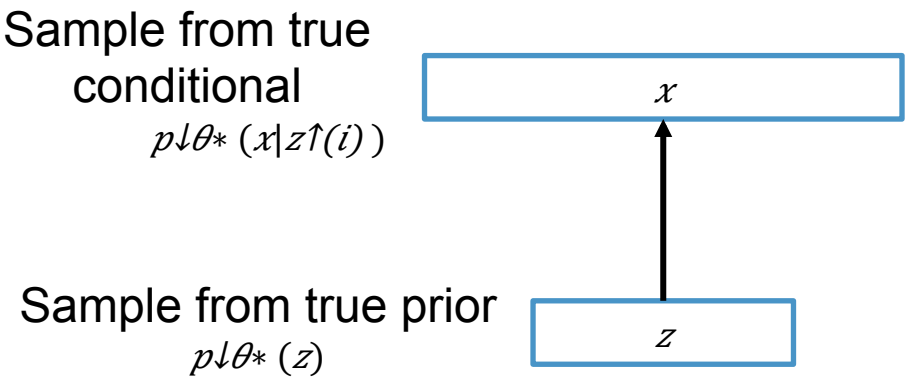
Variational Autoencoders

- Autoencoders can reconstruct data, and can learn features to initialize a supervised model
- Features capture factors of variation in training data. Can we generate new images from an autoencoder?
- Variational Autoencoders(VAE) - Probabilistic spin on autoencoders - will let us sample from the model to generate data!



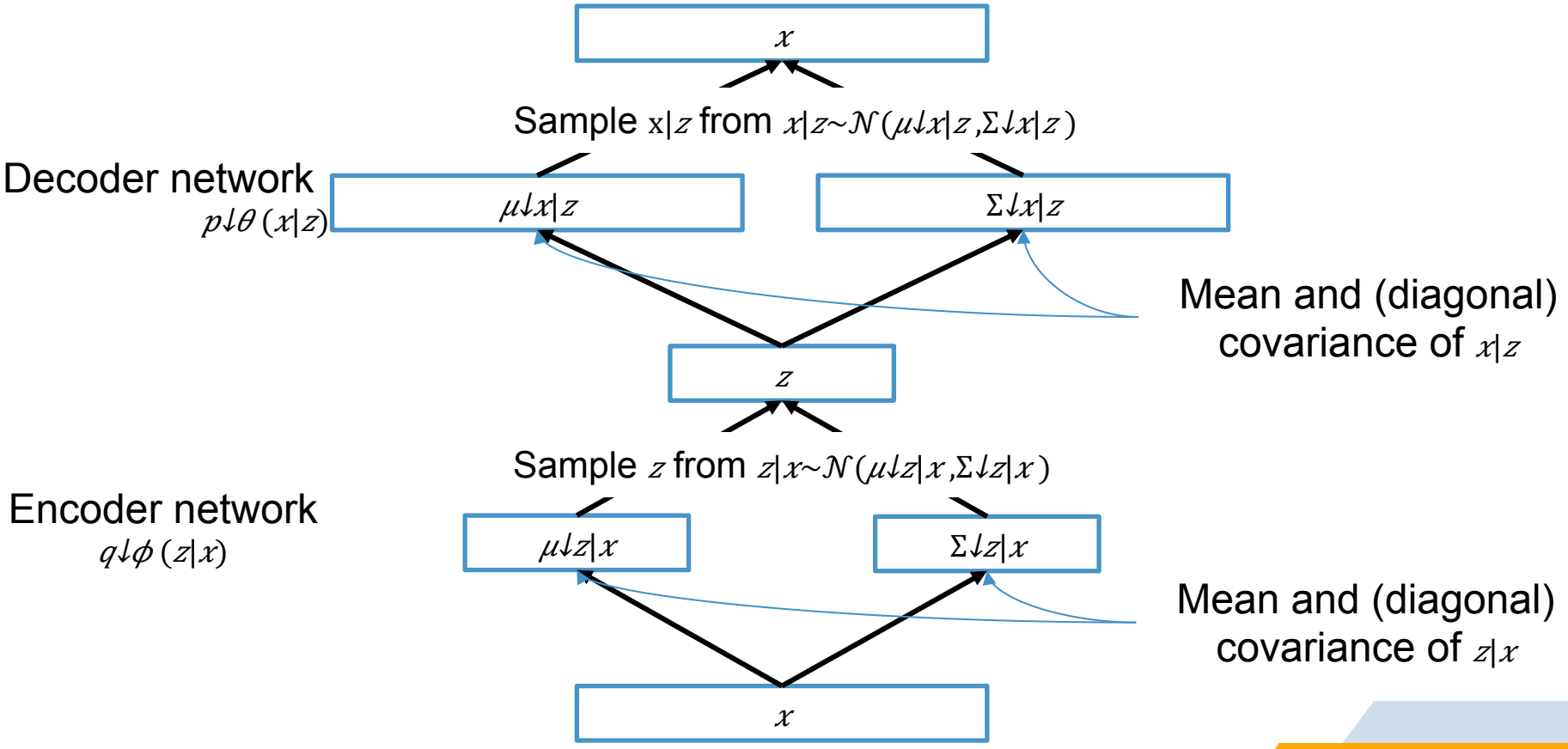
Variational Autoencoders

- Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation z



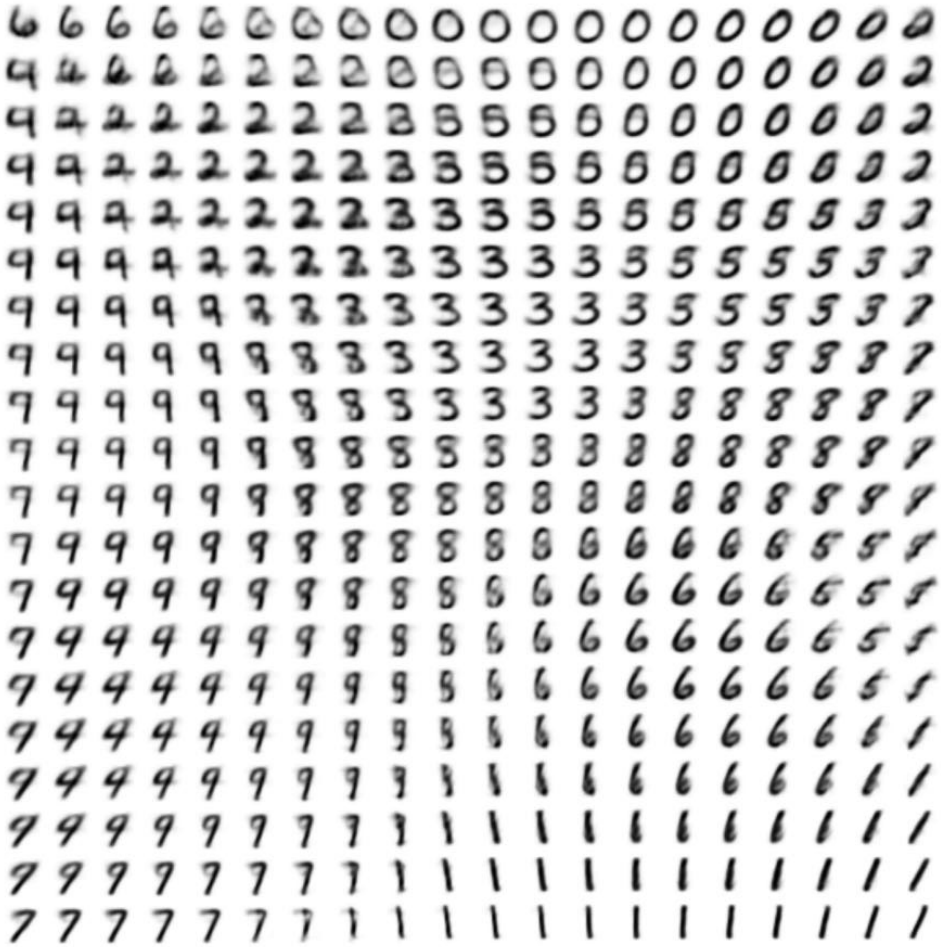
- Intuition: x is an image, z is latent factors used to generate x: attributes, orientation, etc.
- How should we represent this model?
- Choose prior $p(z)$ to be simple, e.g.Gaussian.
- Conditional $p(x|z)$ is complex (generates image) => represent with neural network

Variational Autoencoders



Samples Generated using VAE

(a) Learned Frey Face manifold



(b) Learned MNIST manifold

Variational Autoencoders

- Probabilistic spin to traditional autoencoders => allows generating data
- Defines an intractable density => derive and optimize a (variational) lower bound
- Pros
  - Principled approach to generative models
  - Allows inference of $q(z|x)$, can be useful feature representation for other tasks
- Cons
  - Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
  - Samples blurrier and lower quality compared to state-of-the-art (GANs)
- Active areas of research
  - More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian
  - Incorporating structure in latent variables

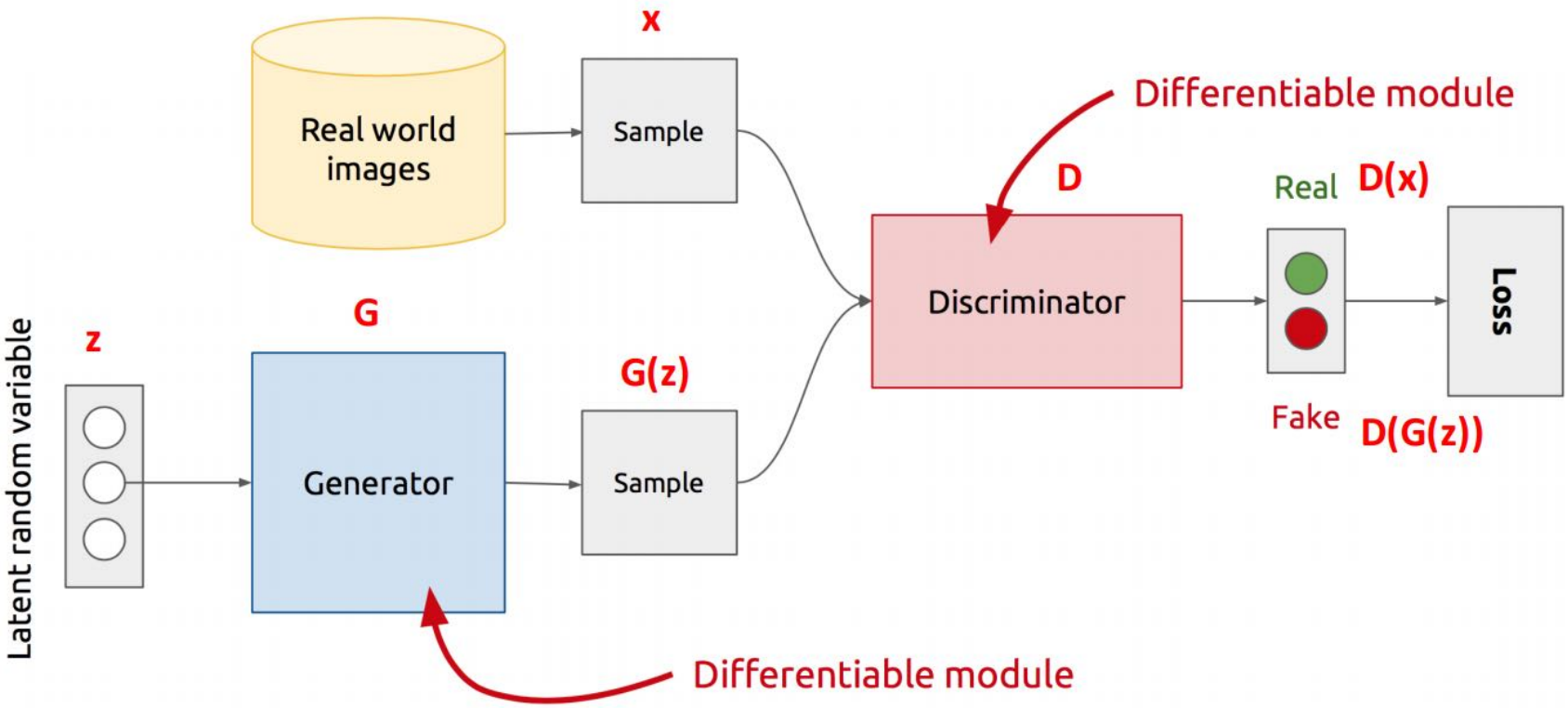
Generative Adversarial Network

- Generative: Learn a generative model
- Adversarial: Trained in an adversarial setting
- Network: Use Deep Neural Networks

Generative Adversarial Network

- GAN is made of a Generator and a Discriminator
  - Generator: generate fake samples, tries to fool the Discriminator
  - Discriminator: tries to distinguish between real and fake samples
  - Train them against each other
  - Repeat this and we get better Generator and Discriminator

GAN's Architecture



GAN Examples

- Image super-resolution

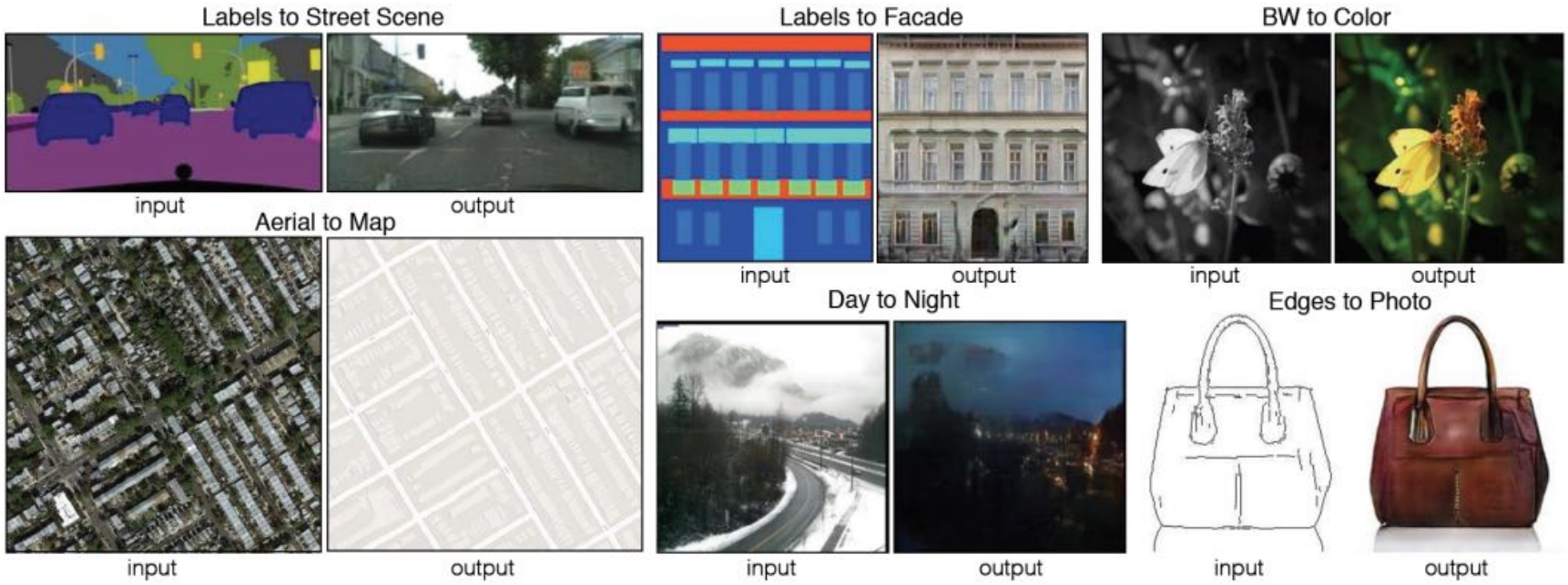
GAN Examples

- Text-to-Image Synthesis

This bird is white with some black on its head and wings, and has a long orange beak	This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face	This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments
		

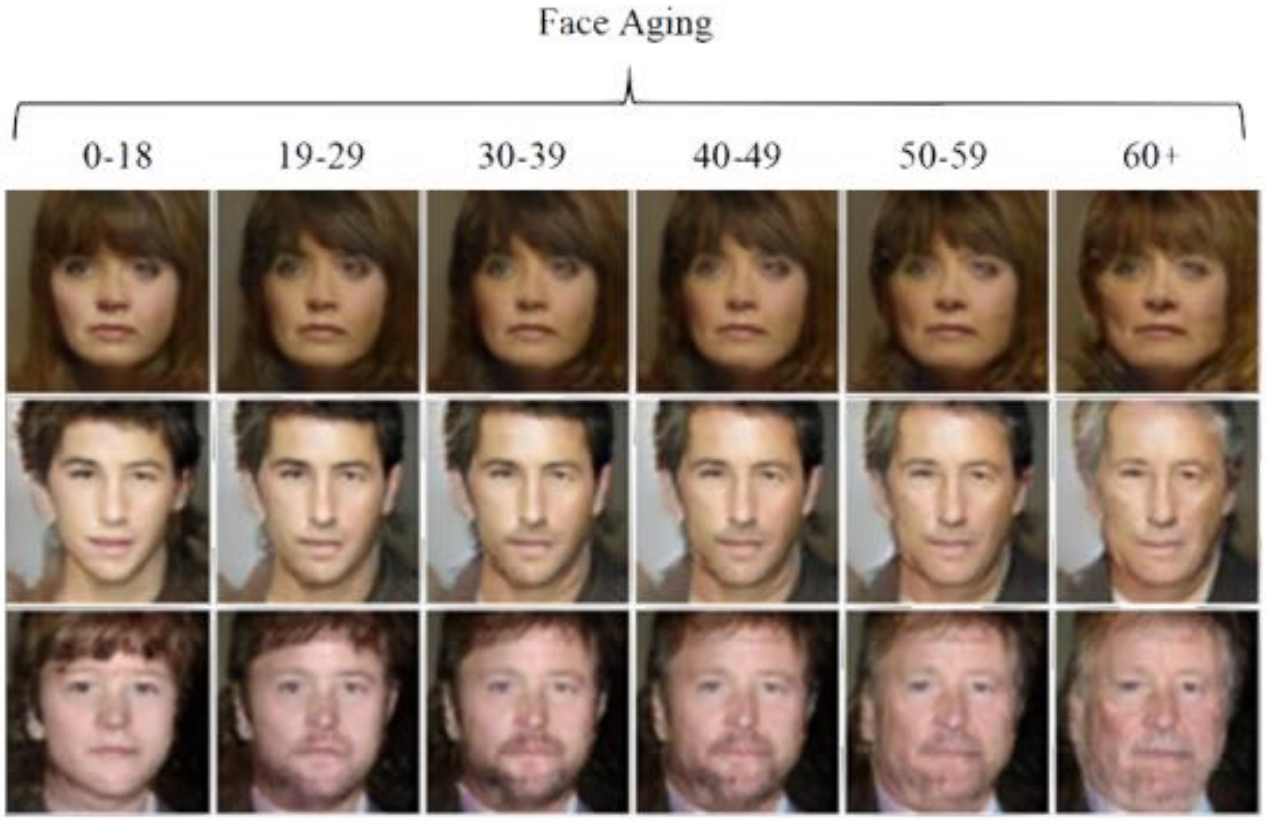
GAN Examples

- Image-to-Image Translation



GAN Examples

- Face aging with conditional GANs



GAN and NLP

- GAN has shown superior results in image generation
- Discreteness in natural language made it hard to apply GAN in NLP
- Recent studies has shown promising results of GAN in NLP(e.g. Boundary-Seeking GAN, Improved W-GAN)

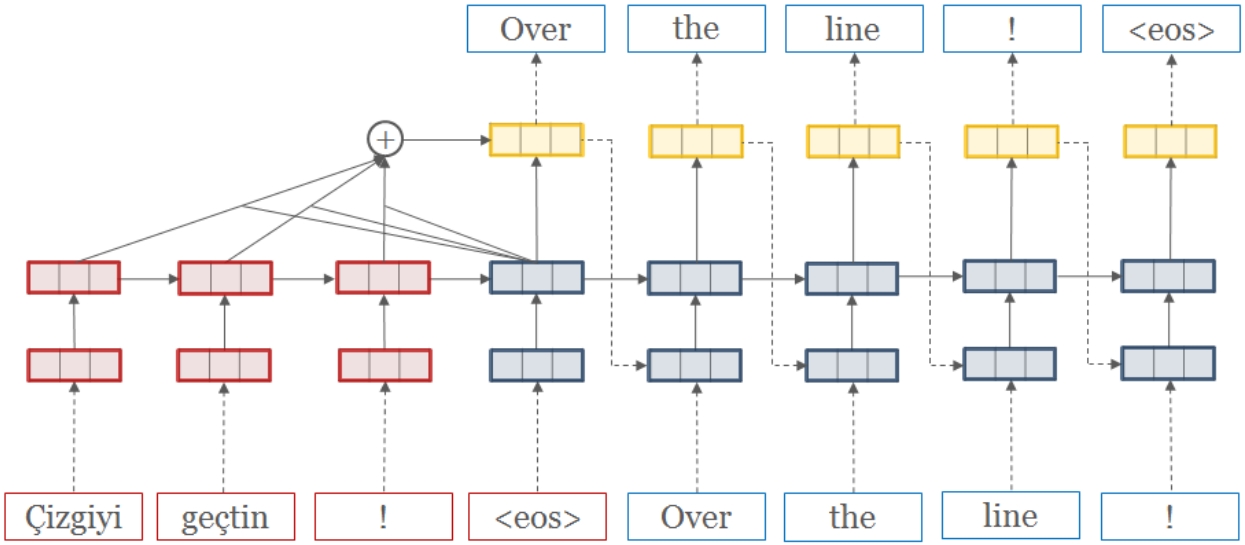
Future Trends

- With distributed representation, various deep models have become the new state-of-the-art methods for NLP problems.
- More NLP applications that employ reinforcement learning and unsupervised learning methods are expected to be seen in the future.
  - Reinforcement Learning:
    Represents a natural way to train NLP systems for the optimization of a particular goal.
  - Unsupervised Learning:
    Promises to learn rich language structure in large unlabeled data.
- Expect to see more deep learning models whose internal memory is enriched with an external memory.
  - Internal memory: Bottom-up knowledge learnt from the data.
  - External memory: Top-down knowledge inherited from a KB.

TensorFlow를 이용한 실습

Word Representation

Neural Network and Language



Human
-> Characters

Neural Network
-> Numbers

Converting Characters to Numbers

“… … … … …. The fat cat sat on the mat. … … … … …“

“… … … … …. 32 832 561 634 6132 32 565 … … … … …”

One-hot Encoding

- V = {cat, fat, mat, sat, the, on}

cat = [1, 0, 0, 0, 0, 0]

fat = [0, 1, 0, 0, 0, 0]

mat = [0, 0, 1, 0, 0, 0]

sat = [0, 0, 0, 1, 0, 0]

the = [0, 0, 0, 0, 1, 0]

on = [0, 0, 0, 0, 0, 1]

- Very high dimensionality, as |V| is typically very large

Limitation of Word-Level Models

- Atomic representations such as one-hot encoding cannot capture the semantics of words

Human: “The fat cat sat on the mat.”

NN: “32 832 561 634 6132 32 565.”

⇨

?!??!

Distributed Representation

Deep Learning based NLP

- Statistical NLP has emerged as the primary option for modeling complex natural language tasks.
- But in the beginning it used to suffer from the ‘Curse of Dimensionality’.

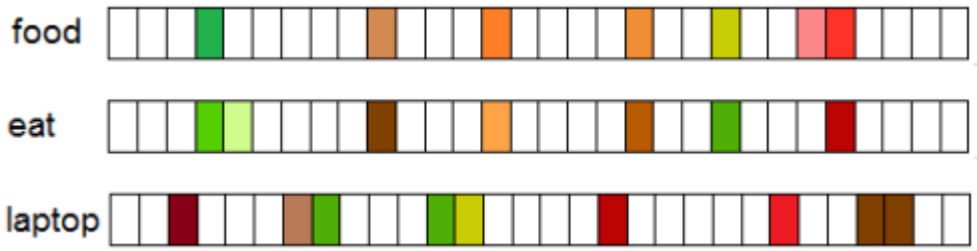
- What is ‘Curse of Dimensionality’?
  - Refers to various phenomena that arise when analyzing and organizing data in high-dimensional spaces which do not occur in low-dimensional settings.
  - The common theme of these problems is that when the dimensionality increases, the volume of the space increases so fast that the available data become sparse.

- This led to the motivation of learning distributed representations of words existing in low-dimensional space.

Distributed Representation

Word Embeddings

- In the past, models that created word embeddings have been shallow neural networks and there has not been need for deep networks to create good embeddings.
- But now Deep learning based NLP models invariably represent their words, phrases and even sentences using word embeddings.
- This marks a major shift in trends between traditional wordcount based models and deep learning based models.



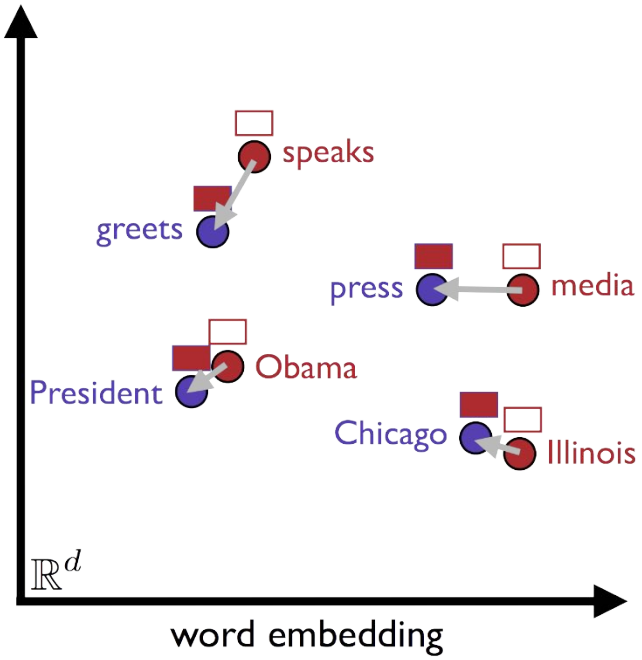
Distributional vectors as a D-dimensional vector where D << V, where V is size of Vocabulary.

What is Word Embedding?

- Mapping sparse word vectors(e.g. one-hot encoding) to a low dimensional dense vector space
- Also known as
  - Distributional Semantic Model
  - Distributional Semantics
  - Distributed Representation
  - Semantic Vector Space
  - Word Space
  - …

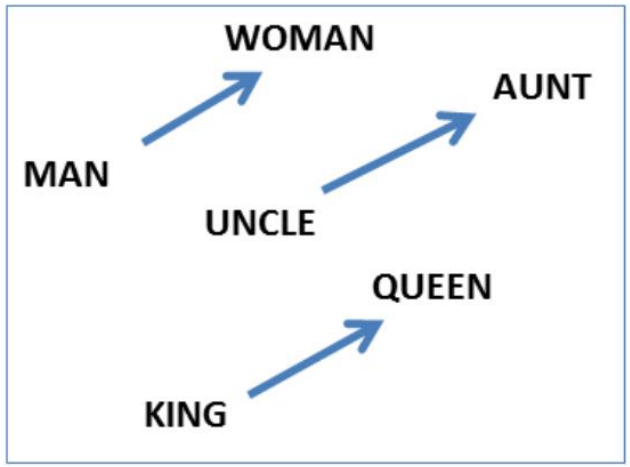
The Goal of Word Embedding

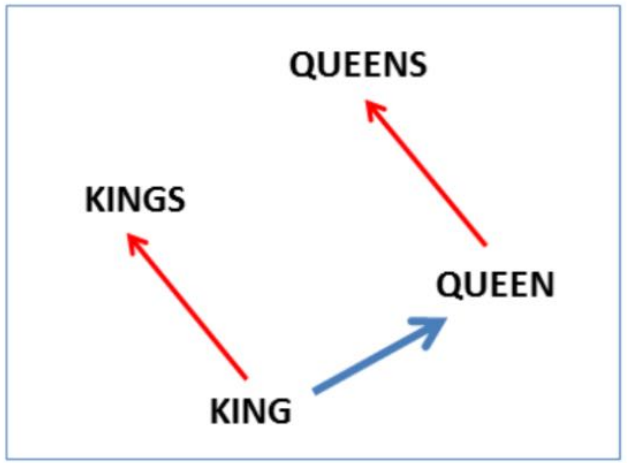
- Map words into low dimensional dense vector space, while preserving semantic relationships
  - Place semantically similar words close to each other



Word Embedding

- Possible to do word analogy task
  - $king - man + woman \approx queen$





Word Embedding Methods

- Hand crafted features (1960s)
- Latent Semantic Analysis(LSA) (1990s)
- Latent Dirichlet Allocation(LDA) (1990s)
- Self Organizing Maps(SOM) (1990s)
- Simple Recurrent Networks(SRN) (1990s)
- Neural Language Model (2000s)
- + Neural network models that accept words as input implicitly create word embeddings

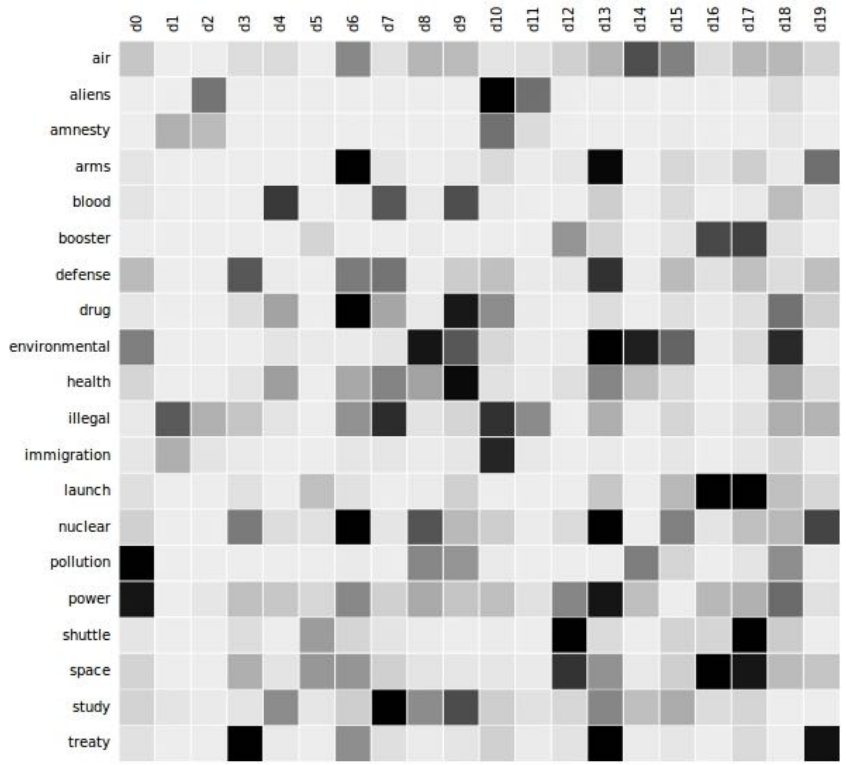
Distributed Representation

Word Embeddings

- During the 1990s, several research developments marked the foundations of research in distributional semantics.
  - Eventually led to the creation of topic models.
- In 2003, Bengio et al. 2003 proposed a neural language model which learnt distributed representations for words.
  - This helped turned out to help generalization since unseen sentences could now gather higher confidence if word sequences with similar words were already seen.
- Collobert and Weston, 2008 established word embeddings as a useful tool for NLP tasks.
  - They They were the first to show the utility of pre-trained word embeddings by proposing a neural network architecture that forms the foundation to many current approaches.
- Mikolov et al. 2013 is responsible for the immense popularity of word embeddings, which showcased the Word2vec.

Latent Semantic Analysis

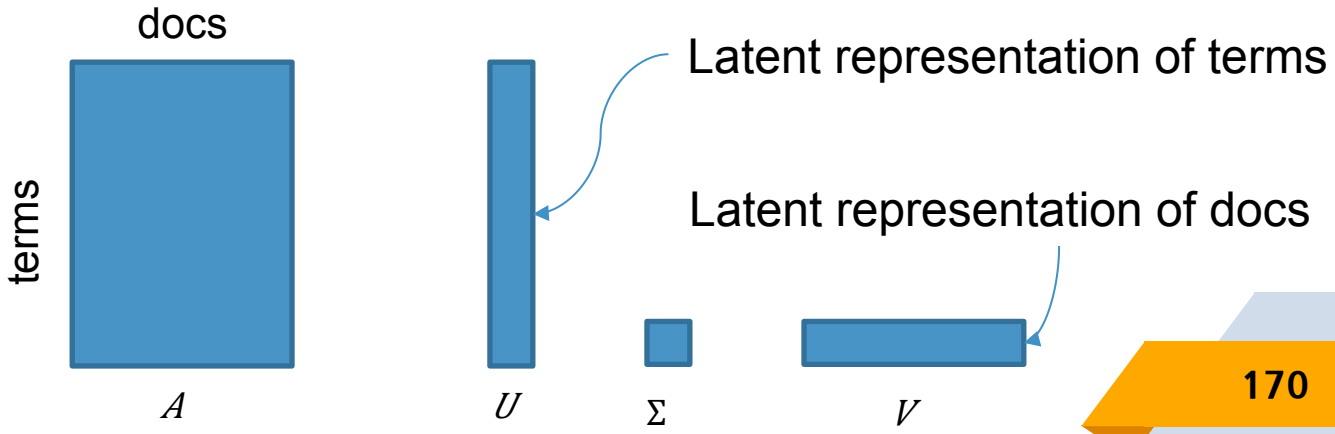
- Decompose a term-document matrix using Singular Value Decomposition(SVD) to obtain a lower-dimensional dense word vector



Term-document matrix

Singular Value Decomposition

- SVD allows us to factorize a matrix as a product of 3 matrices
- $A\downarrow m\times n = U\downarrow m\times r\, \Sigma\downarrow r\times r\, (V\downarrow n\times r)\uparrow T$
- A is the input matrix, U has the left singular vectors, Σ has the singular values, V has the right singular vectors
- It is always possible to decompose a matrix A as a product of U,Σ, V, and these are unique for a given A



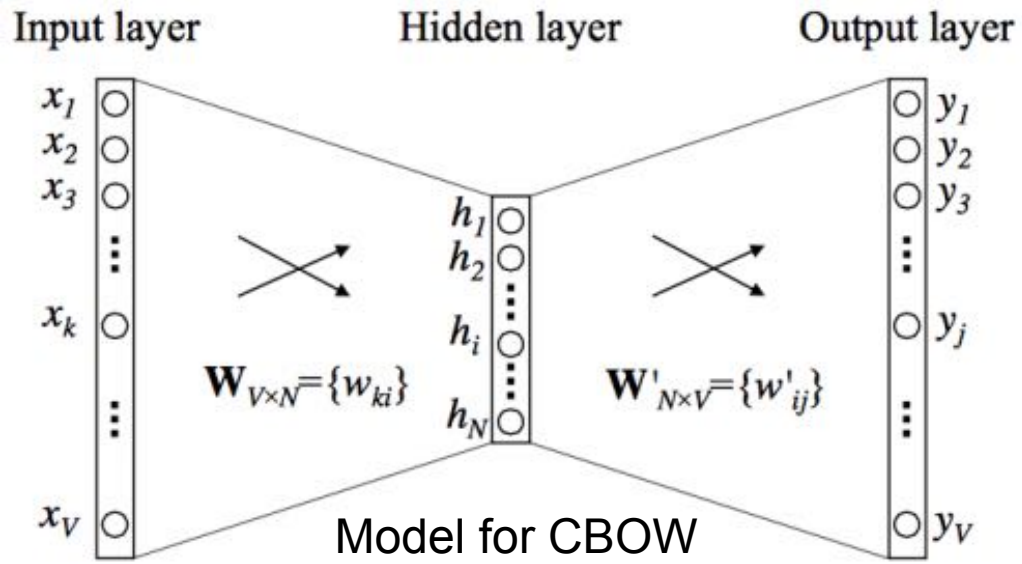
Distributional Hypothesis

- Hypothesis that support Word2Vec and other similar word embedding methods
- “Words that occur in the same contexts tend to have similar meanings”
  - Harris, Z. (1954). Distributional structure. Word, 10(23): 146-162.
- Models that does predictions such as context -> word(CBOW), word -> context(Skip-gram) can implicitly learn the semantics of words

Distributed Representation

Word2vec

- Word Embedding was revolutionized by Mikolov et al. by the proposition of CBOW & skip gram models.
  - CBOW computes the conditional probability of a target word given the context words surrounding it across a window of size k.
  - Skip-gram model does the exact opposite of the CBOW model by predicting the surrounding context words given the central target word.



Model for CBOW

Distributed Representation

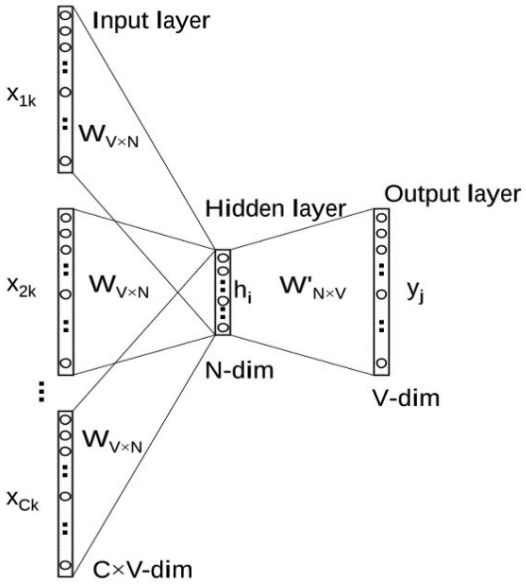
Word2vec

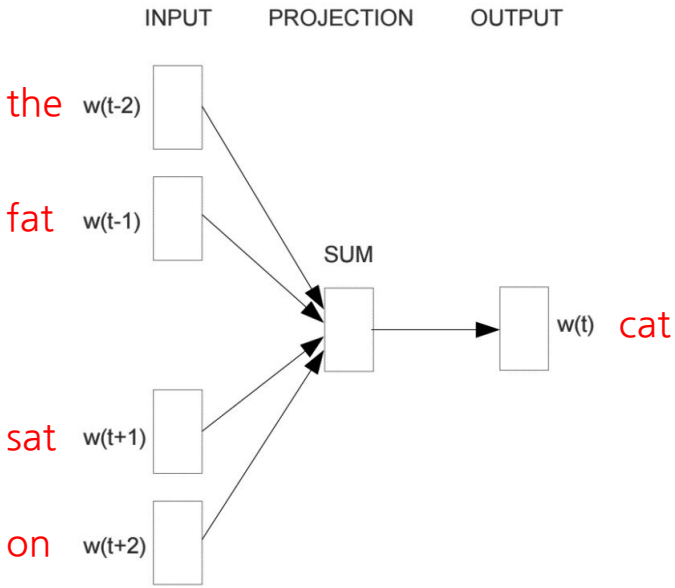
- limitations of individual word embeddings were also pointed out by Mikolov et al.
  - The combination of 2 or more words does not represent the combination of meanings of individual words.
    ex: Hot Potato (Game), Boston Globe (Idiom) etc.
  - Learning embeddings based only on a small window of surrounding words sometimes cluster semantically-similar words which have opposing sentiment polarities.
    → This leads the downstream model used for the sentiment analysis task to be unable to identify this contrasting polarities leading to poor performance.

- Mikolov et al.'s proposition Word2vec has redeemed at least one of those limitations.
  - To identify multi-word combinations based on word co-occurrence while training them for embeddings separately.
    → More recent methods have explored directly learning n-gram embeddings from unlabeled data.

Continuous Bag-Of-Words

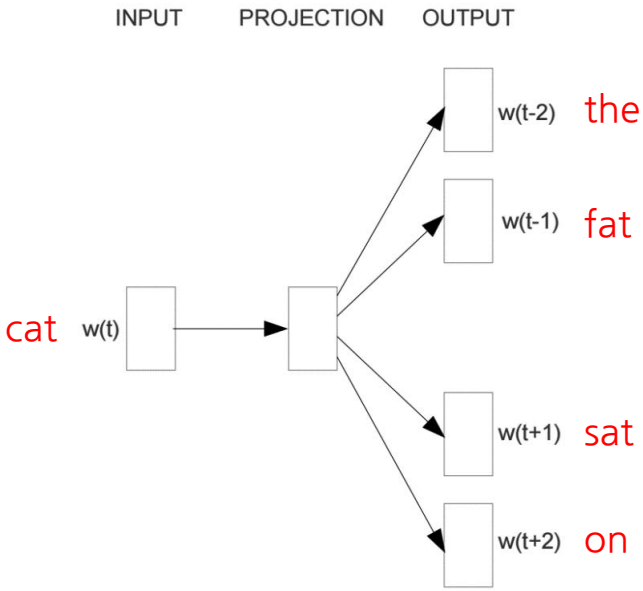
- Predict center word given its context words
  - The fat cat sat on the mat.





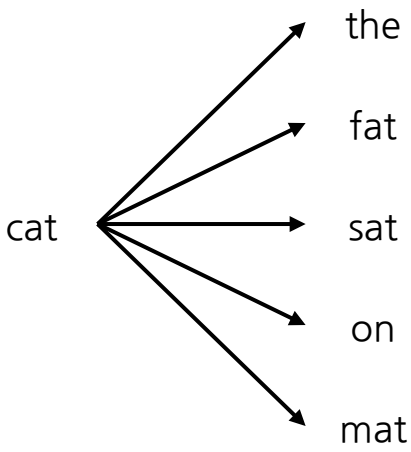
Skip-gram

- Predict context words given its center word
  - The fat cat sat on the mat.

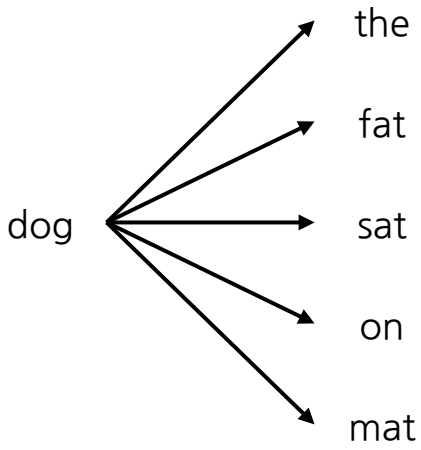


Why Does it Work?

"The fat **cat** sat on the mat"



"The fat **dog** sat on the mat"



cat ≈ dog

Limitations of Skip-Gram

- Hard to learn good representation for rare words
  - Data sparseness problem
- Out-of-Vocabulary words problem
- More problematic in agglutinative languages such as Korean or Turkish
- Integrating sub-word(character-level) information can help to overcome this problem

Word/Morpheme Level Model

- Treat word/morpheme as atomic units
- Limitations
  - Needs a fixed and predefined list of possible words/ morphemes(vocabulary)
  - Number of parameters in NN grows linearly with the number of words/morphemes in vocabulary
  - Causes Out Of Vocabulary(OOV) words

Out-Of-Vocabulary Word

- Cannot have every possible words in vocabulary
  - Computer resources are limited
  - Language constantly evolves, introducing new words
- OOV words are replaced with a special token(e.g. “<unk>”)

Distributed Representation

Character Embeddings

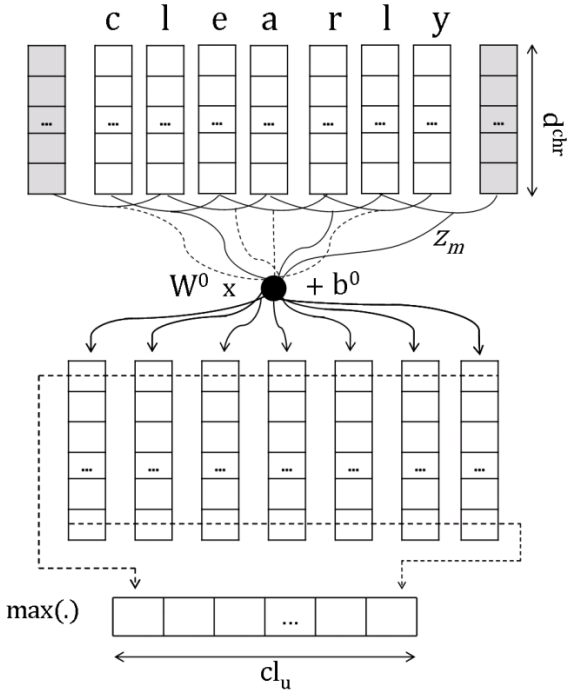
- While Word Embeddings can capture syntactic and semantic information, for NLP tasks such as POS-tagging and NER, intra-word morphological and shape information are also very useful.
- Recently building natural language understanding systems at the character level has been attracting a certain amount of attention.
  - Better performances on morphologically rich languages are reported in certain NLP tasks.
  - Santos and Guimaraes, 2015 applied character-level representations, along with word embeddings for NER, achieving state-of-the-art results in Portuguese and Spanish corpora.
  - Kim et al., 2016 showed positive results on building a neural language model using only character embeddings.
- Character embeddings also naturally deal with the unknown word issue, since each word is considered no more than compositions of individual letters.

Character-Level Model

- Uses characters as atomic units instead of words/ morphemes
- In most languages, the number of characters is much smaller than the number of words
- The set of characters are much less likely to change
- Mainly two approaches
  - Convolution
  - Character-level Recurrent Neural Network

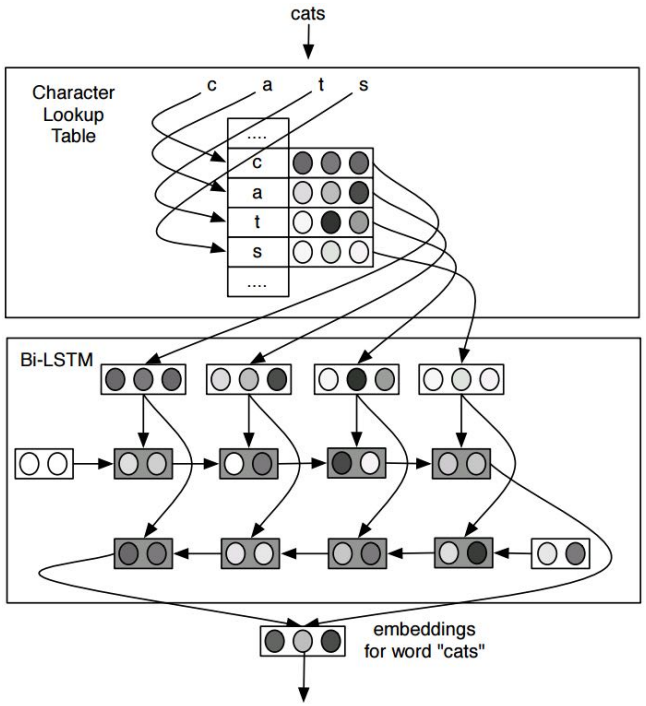
Convolution

- Convert each character to character embedding in the same way as word embedding
- Convolution is applied to the character embedding of the characters constituting the word, and word vectors are generated using pooling



Character RNN

- Generate word vectors by applying RNN to the character embedding of the characters constituting the word
- Can be used with or without word embedding



Limitations

- Convolution – ignores character order: confusion with anagrams (altitude/latitude, silent/listen)
- RNN – The number of steps increases (# of words -> # of characters): long term dependency problem
- Introduces additional parameters in model for CNN/RNN

GloVe: Global Vectors for Word Representation

- Motivation of GloVe
  - Word2Vec poorly utilize the statistics of the corpus - calculation efficiency and word representation performance can be improved by using theses statistics
  - Count-based vectors are less universal
  - GloVe: Combines information from Word co-occurrence matrix and context word prediction model

GloVe Highlights

- Good performance even when the amount of data is insufficient or the size of the vector is small
  - Closest words to the target word *frog:*

frogs, toad, litoria, leptodactylidae, rana, lizard, eleutherodactylus

3. litoria

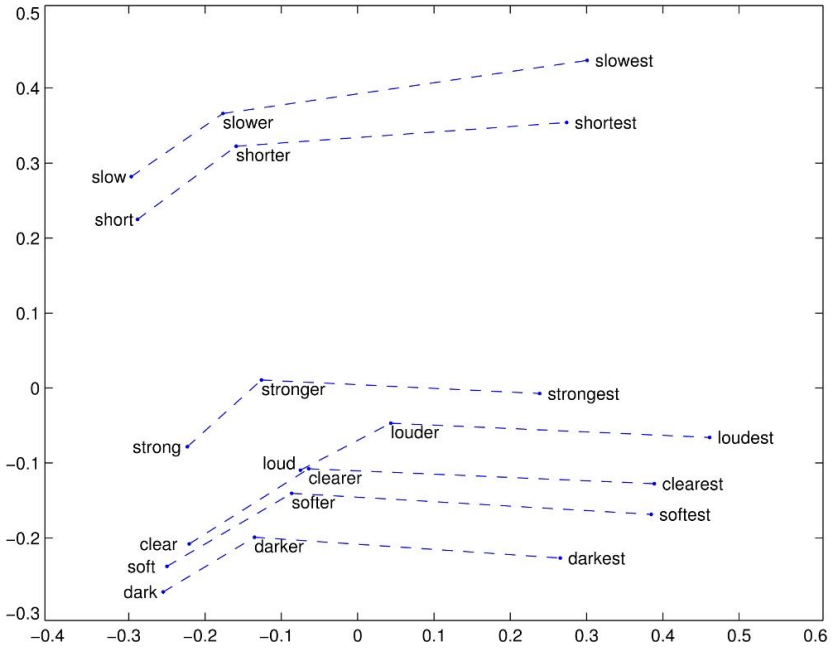
4. leptodactylidae

5. rana

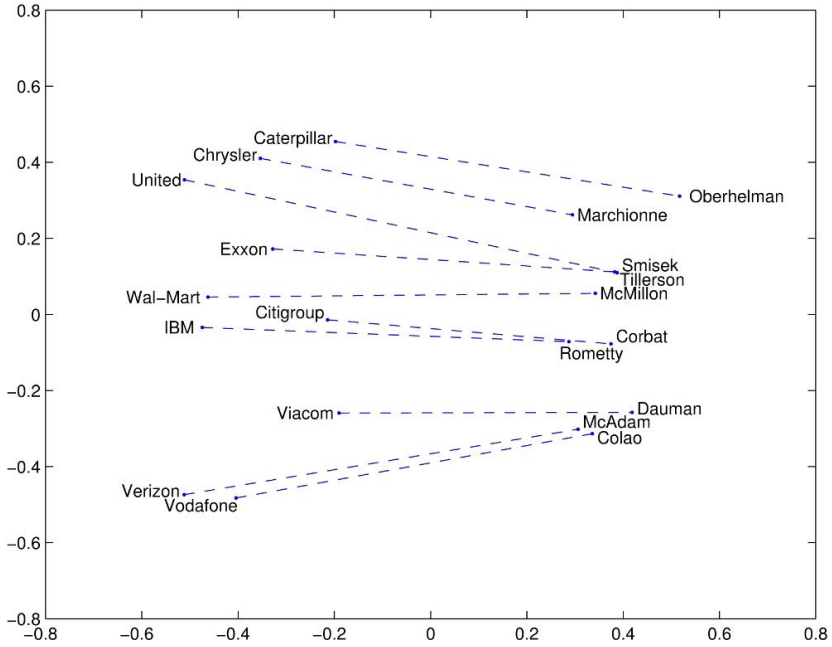
7. eleutherodactylus

GloVe Highlights

- Better word analogy task performance



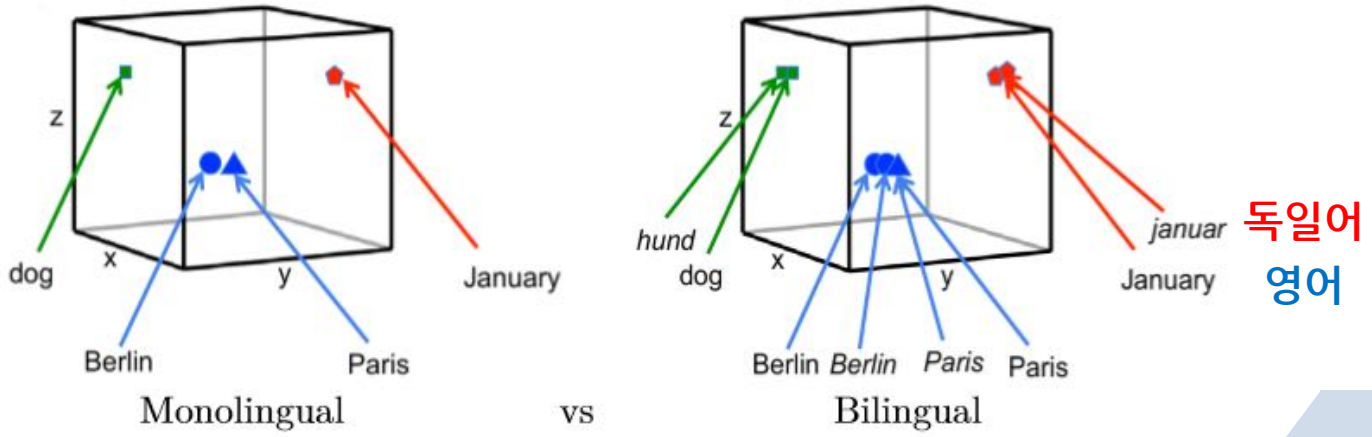
comparative - superlative



company - ceo

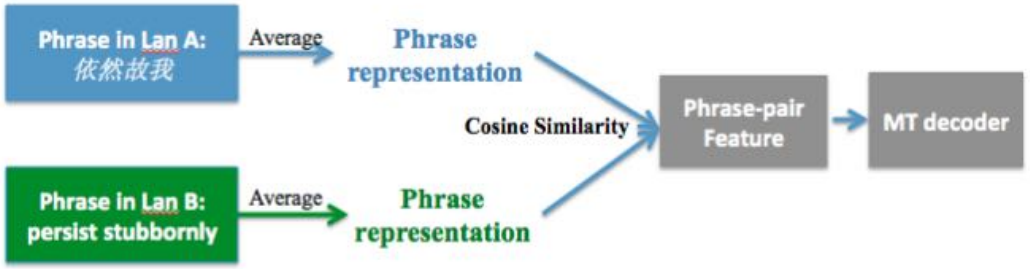
Emerging of Bilingual Word Embedding

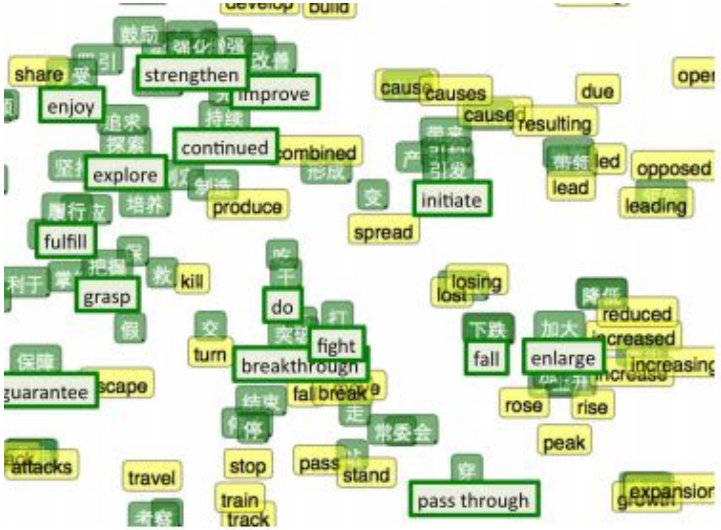
- If you understand the meaning of words through word embedding for a single language (Korean), what about embedding two different languages?
  - Bilingual Word Embedding : Embedding a word in one vector space so that similar words in two different languages are in a similar space



Application of Bilingual Word Embedding

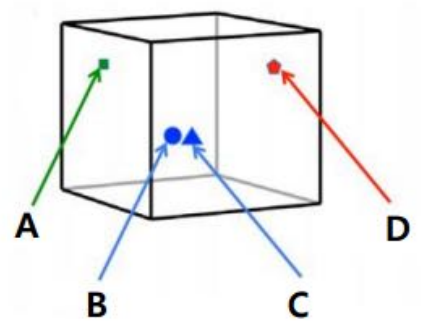
- Machine translation (Zou et al., 2013)





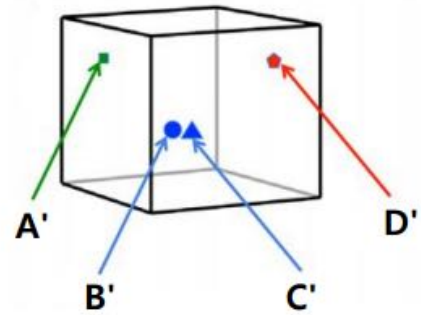
Overview of building a bilingual word embedding

영어



Source WES

한국어



Target WES

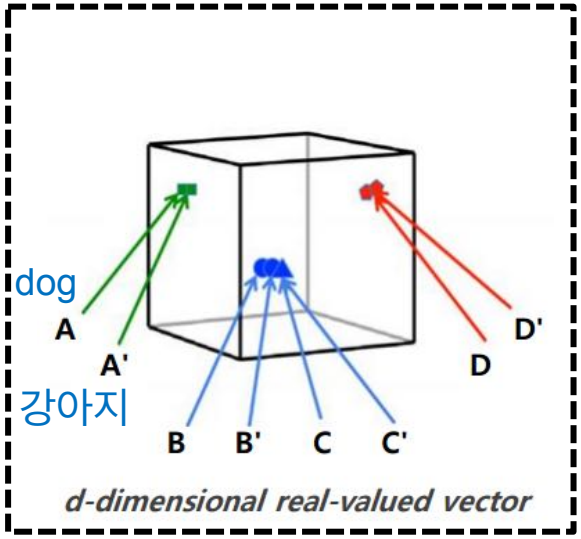
V^S = A, B, C, D

V^T = A', B', C', D'

Small scale bilingual word embedding space

Using seed lexicon

Large scale bilingual word embedding space

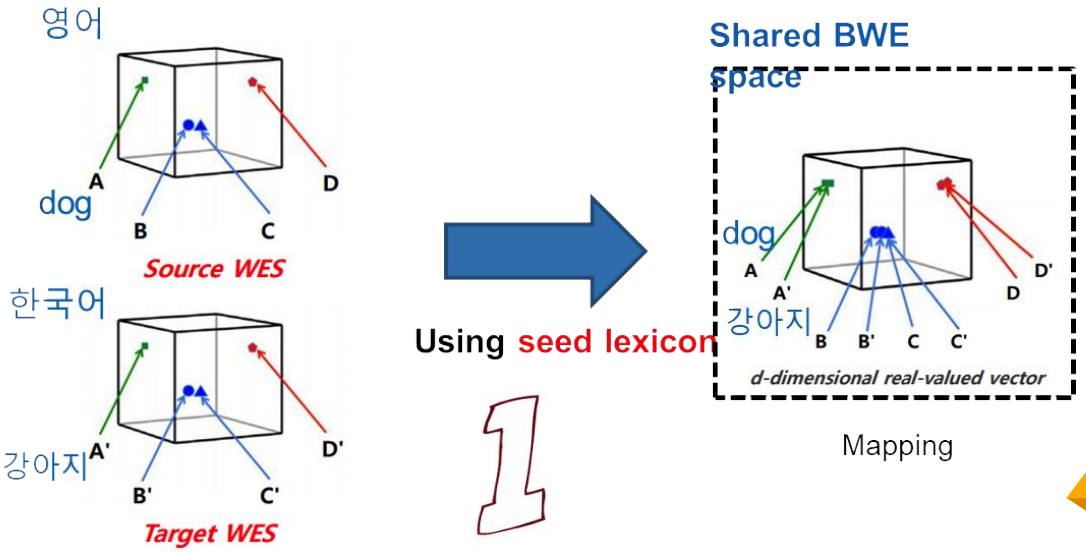


Mapping

유사한 의미의 단어를 유사한 공간에 mapping

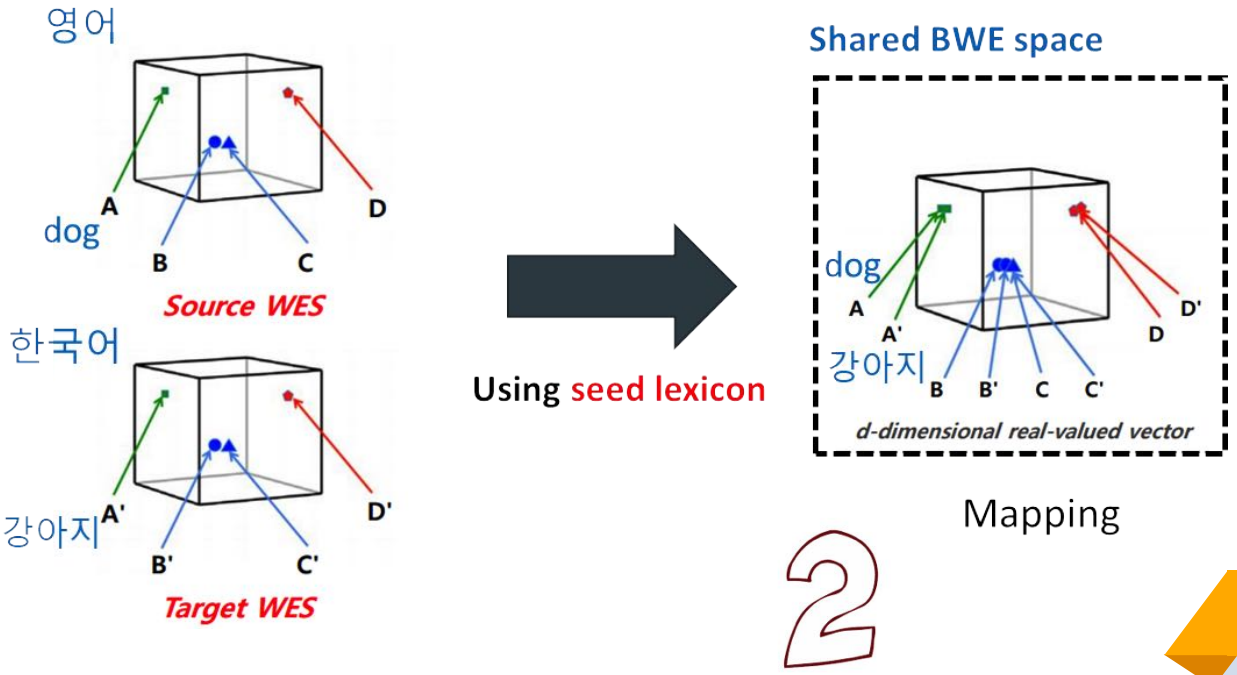
Building a bilingual word embedding (1/2)

- General steps for building a bilingual word embedding
- **Step 1.** Two separately prepared non-aligned monolingual embedding spaces are derived using the monolingual word embedding learning model.
  - Using relatively easy-to-obtain document translation pairs(Wikipedia) as word translation pairs

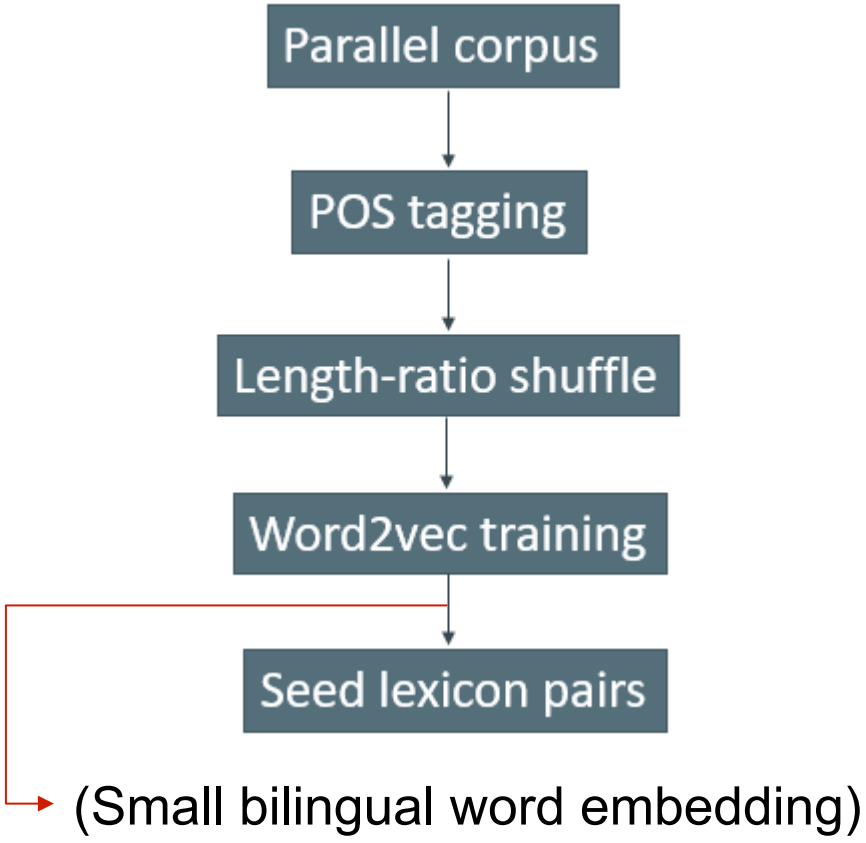


Building a bilingual word embedding (2/2)

- **Step2.** Using the word translation pair as a seed lexicon, we construct a shared BWE(bilingual word embedding) space by learning a mapping function that combines two monolingual spaces together.

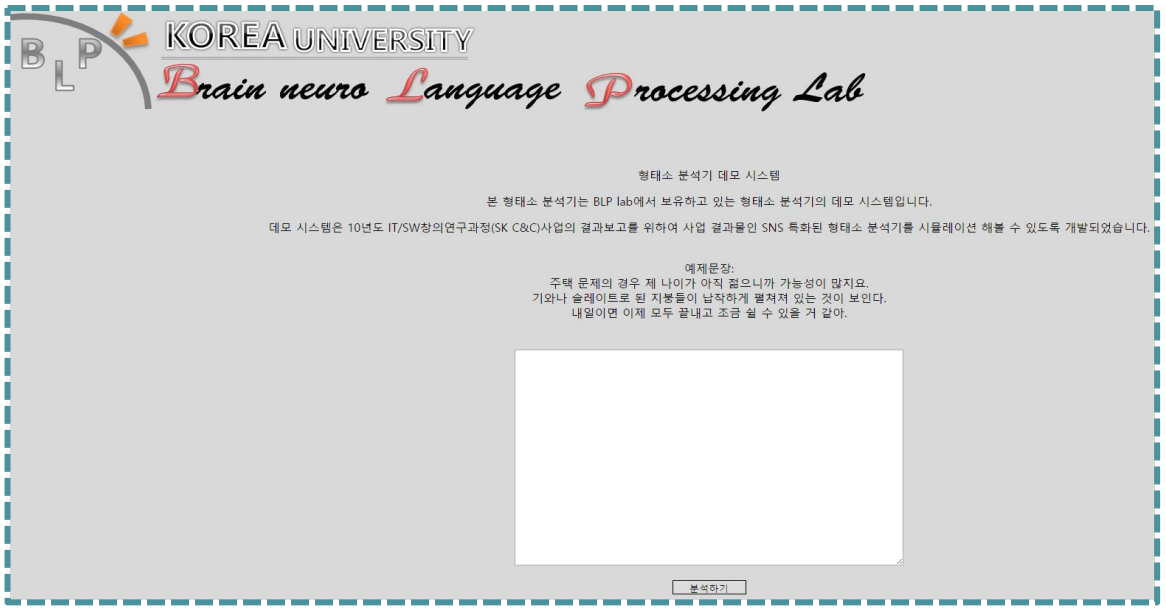


Overview of making the seed lexicons (small bilingual word embedding)



Making the seed lexicon (1/3)

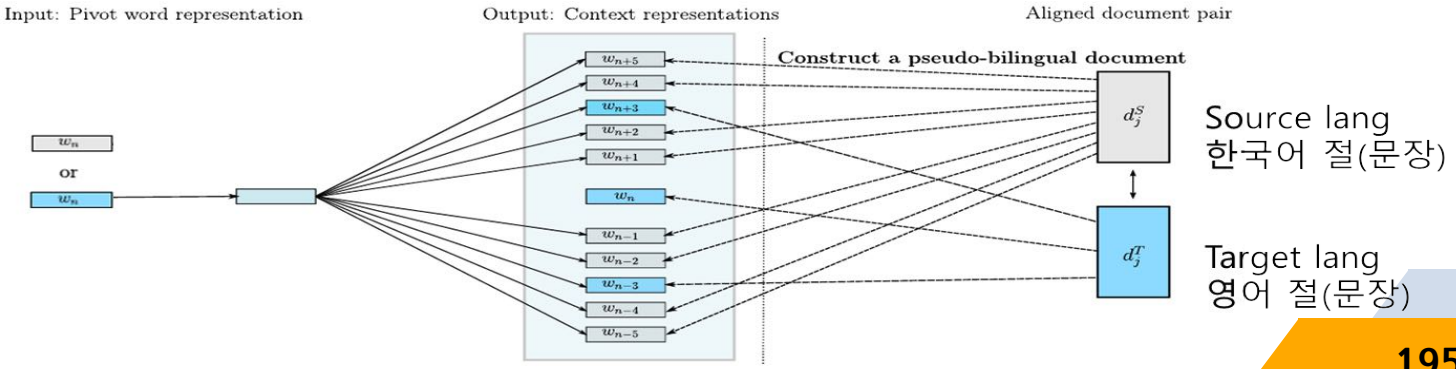
- Preprocessing



한국어 형태소 분석기

Making the seed lexicon (2/3)

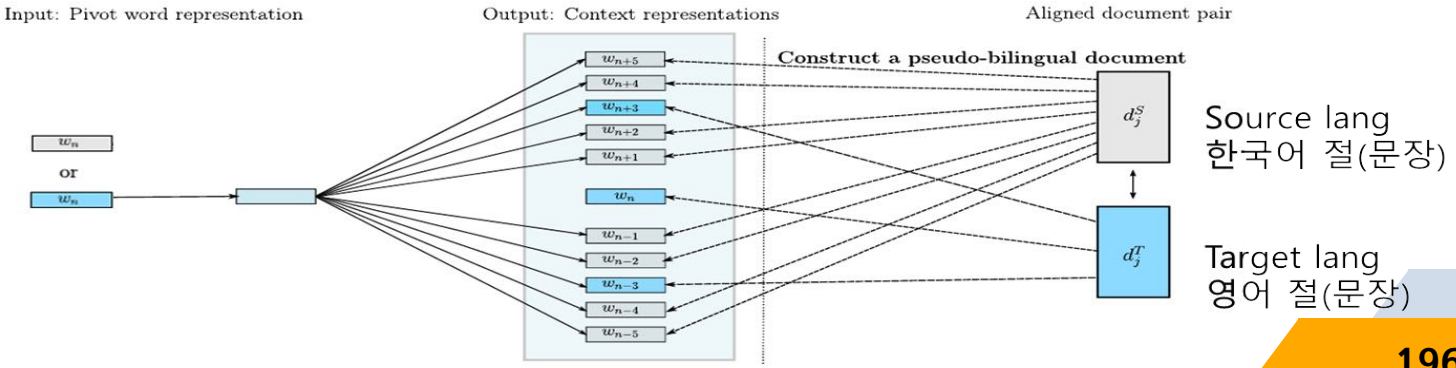
- Length-ration shuffle (vulic, 2015)
  - Sentence alignment In the corpus, the sentences of two different languages are mixed together
  - The ration of the number of tokens per sentence to the shuffle
  - ex) Ko = {포도, 복숭아}
    - Token number = 2
  - ex) Eng = {carrot, apple, pineapple, egg}
    - Token number = 4
    - Length-ratio = 2 : 1



Length-ratio shuffle

Making the seed lexicon (3/3)

- Length-ration shuffle (vulic, 2015)
  - Sentence alignment In the corpus, the sentences of two different languages are mixed together
  - The ration of the number of tokens per sentence to the shuffle
  - ex) Ko = {포도, 복숭아}
    - Token number = 2
  - ex) Eng = {carrot, apple, pineapple, egg}
    - Token number = 4
    - Length-ratio = 2 : 1



Length-ratio shuffle

SGNS

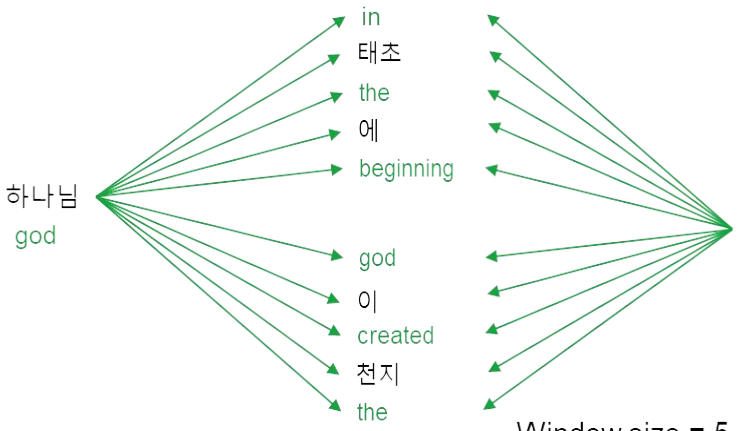
Building a small bilingual word embedding

```
01창 1:1 태초에 하나님이 천지를 창조하시니라
01창 1:2 땅이 혼돈하고 공허하며 흑암이 깊음 위에 있고 하나님의
01창 1:3 하나님이 이르시되 빛이 있으라 하시니 빛이 있었고
01창 1:4 빛이 하나님이 보시기에 좋았더라 하나님이 빛과 어둠을
01창 1:5 하나님이 빛을 낮이라 부르시고 어둠을 밤이라 부르시니라
01창 1:6 하나님이 이르시되 물 가운데에 궁창이 있어 물과 물로
01창 1:7 하나님이 궁창을 만드사 궁창 아래의 물과 궁창 위의 물
01창 1:8 하나님이 궁창을 하늘이라 부르시니라 저녁이 되고 아침이
```

Korean

```
01Gn 1:1 In the beginning God created the heaven
01Gn 1:2 Now the earth was formless and empty,
01Gn 1:3 And God said, "Let there be light," and
01Gn 1:4 God saw that the light was good, and he
01Gn 1:5 God called the light "day," and the dar
01Gn 1:6 And God said, "Let there be an expanse
01Gn 1:7 So God made the expanse and separated
01Gn 1:8 God called the expanse "sky." And there
01Gn 1:9 And God said, "Let the water under the
```

English



Window size = 5

```
in 태초 the 에 beginning 하나님 god 이 created 천지 the 를 heavens 창조 and 하 the earth 시
땅 now 이 the 혼돈 earth 하 was 고 formless 공허하 and 며 empty 흑암 darkness 이 was 깊 over
하나님 and 이 god 이르 시 said 되 let 빛 이 there 있 be 으라 하 light 시 and 니 빛 there 이 was
빛 god 이 saw 하나님 that 이 the 보 light 시 기 was 에 good 좋 and 았 he 더라 separated 하나님
하나님 god 이 called 빛 을 the 낮 light 이 라 day 부르 and 시 the 고 어둠 darkness 을 he 밤 이
하나님 and 이 god 이르 said 시 되 let 물 there 가운데 be 에 an 궁창 이 expanse 있 between 어 th
하나님 so 이 god 궁창 made 을 the 만들 expanse 사 and 궁창 separated 아래 the 의 water 물 unde
하나님 god 이 called 궁창 을 the 하늘 expanse 이 라 sky 부르 and 시 니라 there 저녁 was 이 even
하나님 and 이 god 이르 said 시 let 되 the 천하 water 의 under 물 the 이 sky 한 be 곳 gathered
```

Length-ratio shuffle

Building a large bilingual word embedding (vocabulary expansion)

L2-regularized least-squares error objective

Parallel corpus

POS tagging

Length-ratio shuffle

Word2vec training

Seed lexicon pairs

$$\min_{\mathbf{W} \in \mathbb{R}^{d_S \times d_T}} \|\mathbf{XW} - \mathbf{Y}\|_F^2 + \lambda \|\mathbf{W}\|_F^2$$

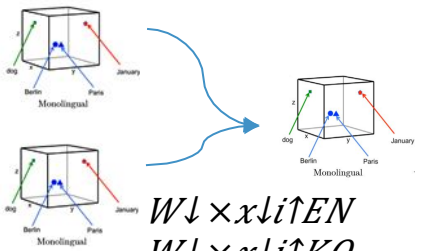
W : 2차원 행렬

X : Source 언어의 word 벡터

Y : Target 언어의 word 벡터

X, Y : Seed lexicon

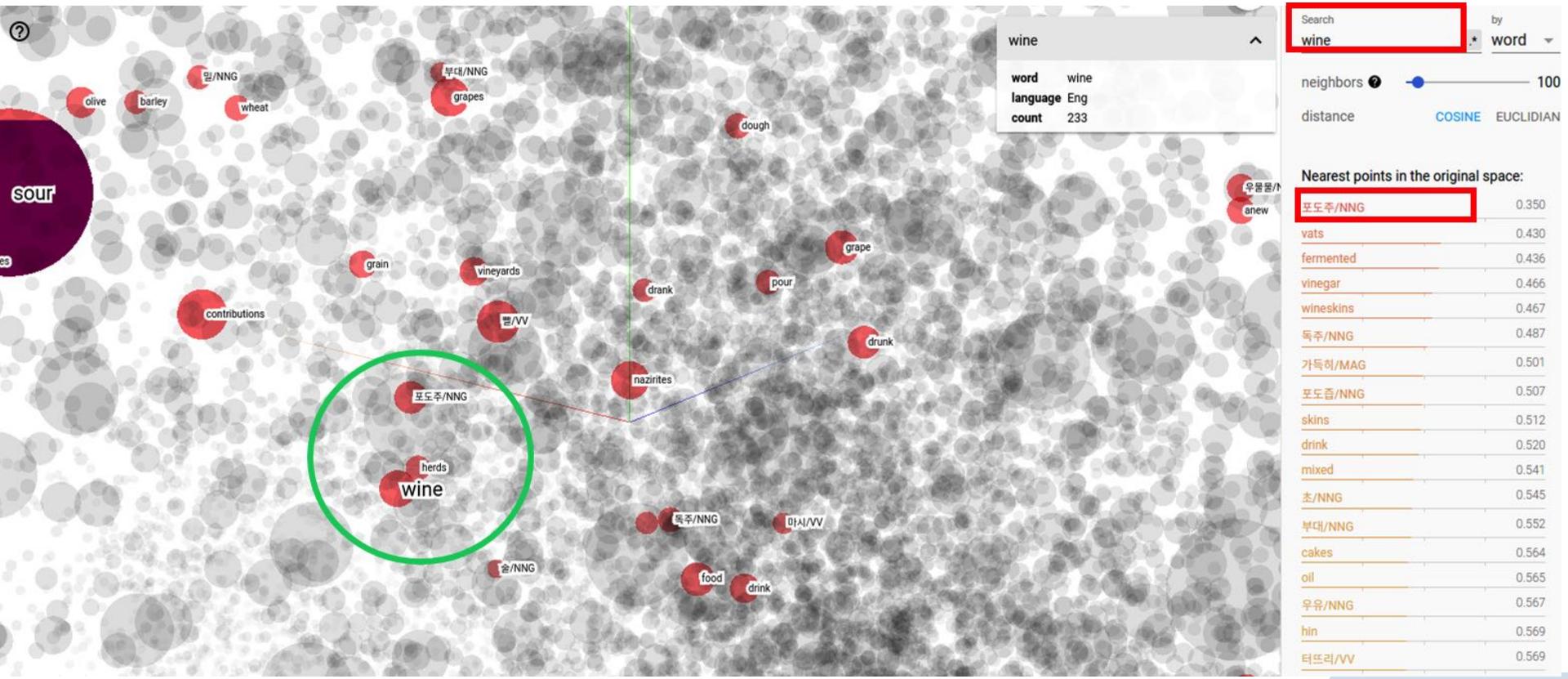
Mapping function



$W\downarrow \times x\downarrow i\uparrow EN$

$W\downarrow \times x\downarrow i\uparrow KO$

Visualization of BWE

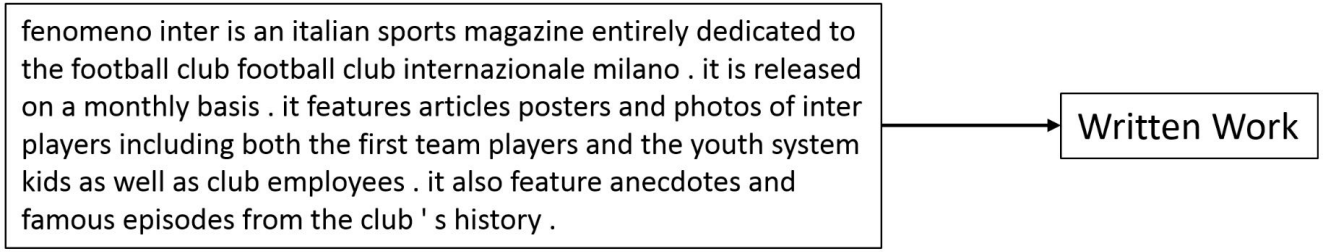


FastText

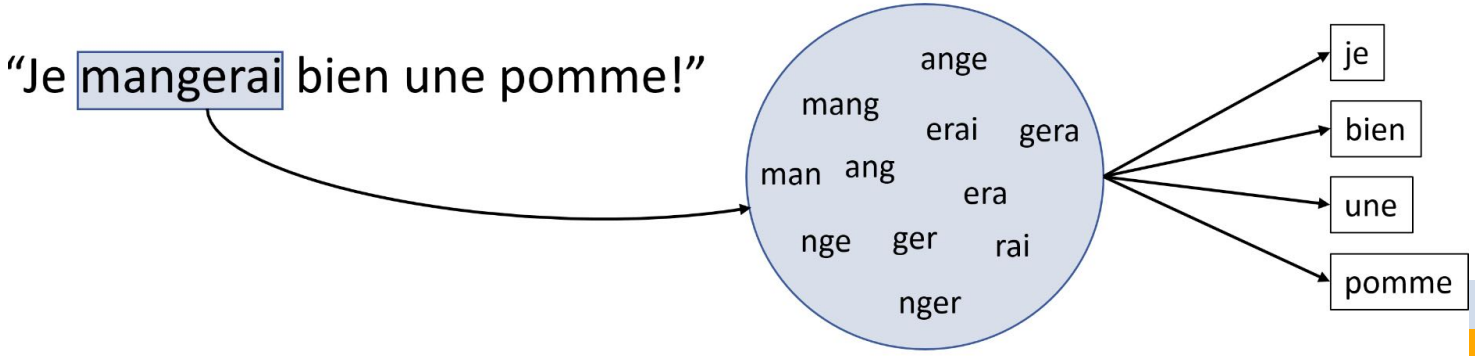
- A library for efficient text classification and word representation
  - Facebook AI Research, 2016
- Unified framework for
  1. Text representation
  2. Text classification

FastText - Two main applications

- Text classification



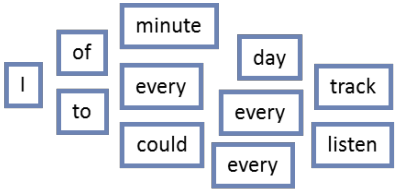
- Word representation (with character-level features)

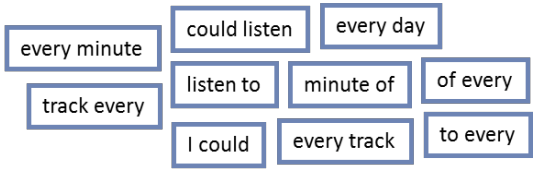


N-grams Feature for Text Classification

- Possible to add higher-order features

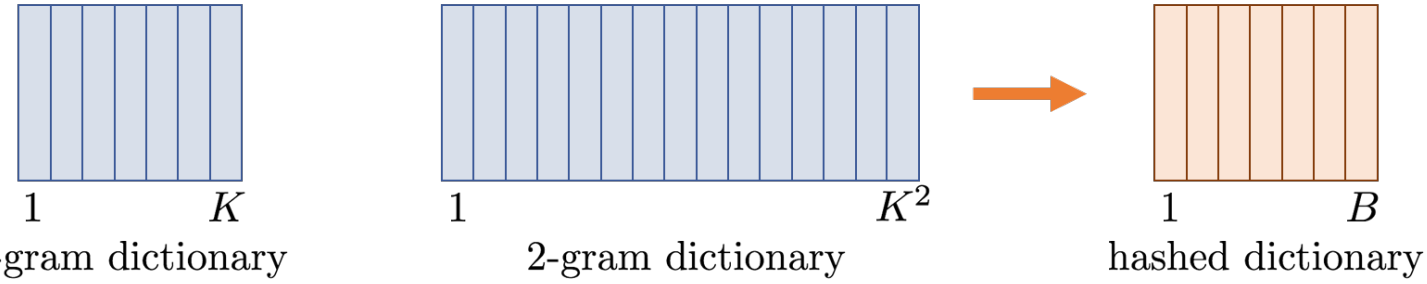
I could listen to every track every minute of every day.





- Avoid building n-gram dictionary

Use a hashed dictionary!



Exploiting sub-word information

- Represent words as sum of its character n-grams
- Add special positional characters: ^mangerai$
- All ending n-grams have special meaning
- Grammatical variations still share most of n-grams

	Singular	Plural
Nominative	uniwersytet	uniwersytety
Genetive	uniwersytetu	uniwersytetów
Dative	uniwersytetowi	uniwersytetom
Accusative	uniwersytet	uniwersytety
Instrumental	uniwersytetem	uniwersytetami
Locative	uniwersytecie	uniwersytetach
Vocative	uniwersytecie	uniwersytety

Polish declension

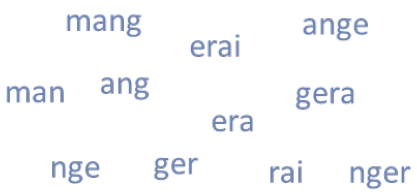
- Compound nouns are easy to model

Tisch Tennis Tischtennis

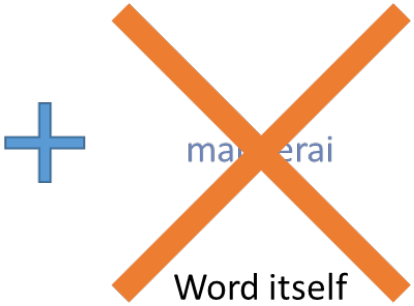
OOV words

- Possible to build vectors for unseen words!

$$h_w = \sum_{g \in w} x_g$$



Character n-grams



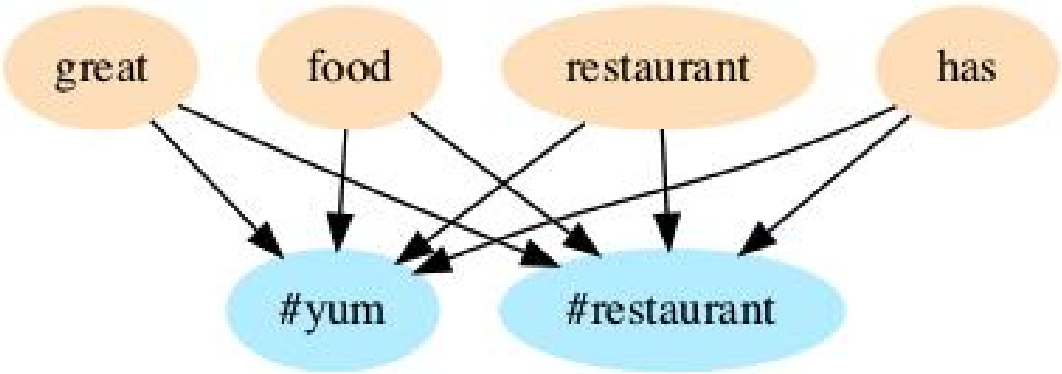
Word itself

StarSpace

- A general-purpose neural model for efficient learning of entity embeddings for solving a wide variety of problems(Facebook AI Research, 2017):
  - Learning word, sentence or document level embeddings
  - Information retrieval: ranking of sets of entities/documents or objects, e.g. ranking web documents
  - Text classification, or any other labeling task
  - Metric/similarity learning, e.g. learning sentence or document similarity
  - Content-based or Collaborative filtering-based Recommendation, e.g. recommending music or videos
  - Embedding graphs, e.g. multi-relational graphs such as Freebase

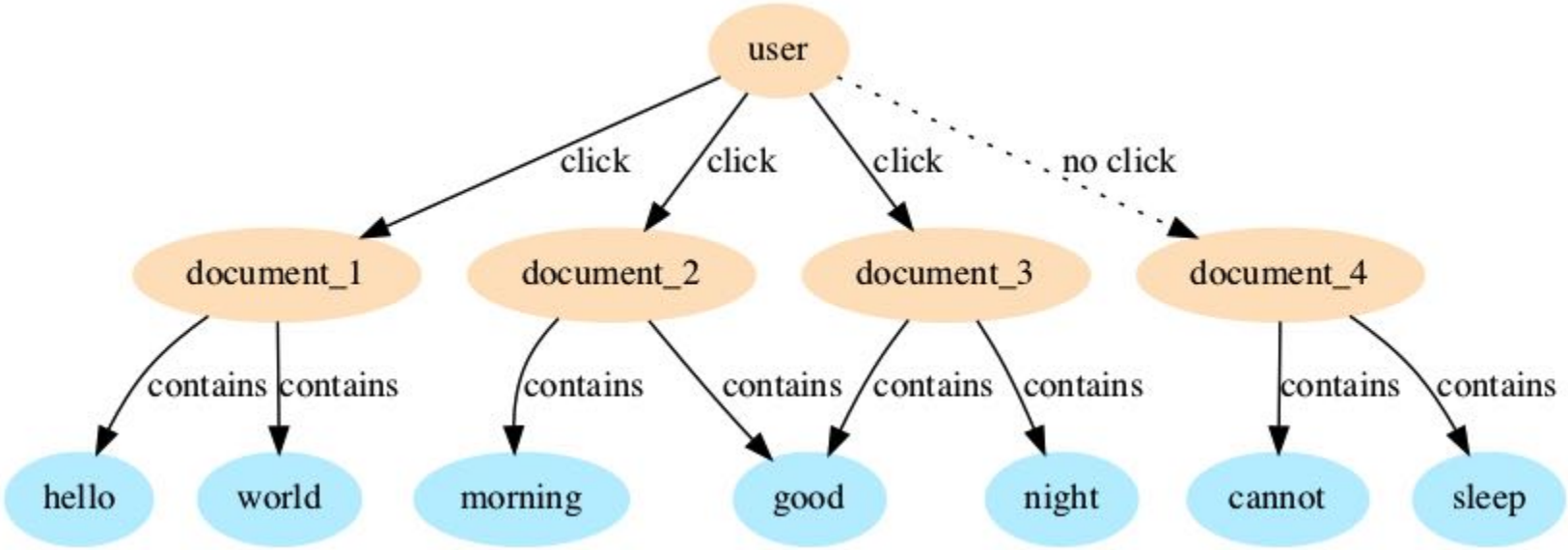
StarSpace Examples

- TagSpace - Learning the mapping from a short text to relevant hashtags
- A classical classification setting.



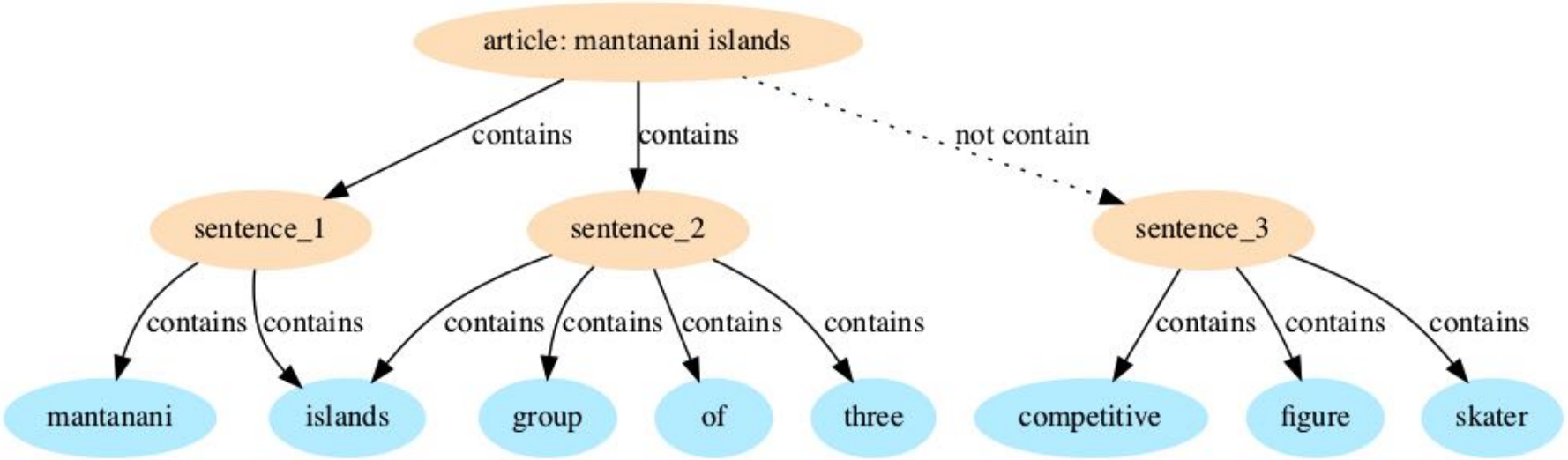
StarSpace Examples

- DocSpace - Embed and recommend web documents for users based on their historical likes/click data



StarSpace Examples

- SentenceSpace - Learning the mapping between sentences. Given the embedding of one sentence, one can find semantically similar/relevant sentences.



Word Representation for Korean

- Most NLP studies are based on English, where words are separated by space
- Korean is a highly agglutinative language, thus word level modeling causes negative effects(e.g. data sparseness problem)
  - [먹다, 먹고, 먹니, 먹지, 먹으며, 먹어서] all shares the same root “eat”, but word level modeling treats each word unique
- Solution: sub-word level modeling(character or morpheme)

Character-Level Model

- Use character-level CNN/RNN to model words
- Sub-character-level modeling is also possible in Korean(consonant and vowel level, or *jaso*)
  - Character-level: 한글 -> 한 + 글
  - Jaso-level: 한글 -> ㅎ + ㅏ + ㄴ + ㄱ + ㅡ + ㄹ

Morpheme-Level Model

- Decompose word into morphemes using morphological analyzer, model words in morpheme-level
- E.g. “가방에 들어가신다” -> 가방 + 에 + 들어가 + 시 + ㄴ다
- Limitations
  - Requires a morphological analyzer
  - Errors in the morphological analyzer can propagate to the downstream tasks

Difficulties of Korean morphological analysis

- Korean is
  - **A highly agglutinative languages**
    - An Eojeol is composed of one or more combined morphemes
  - **A morphologically complex language**
    - Korean words (or Eojeols) are formed through compounding and derivation
    - Also morphological changes are frequently observed
      - “날/Nal/verb”+”는/Neun/connective_ending” → “나는/NaNeun”

Difficulties of Korean morphological analysis

- Korean is
  - **Very productive**
    - The number of Eojeols appeared in real texts is almost infinite
  - **Hard to find the boundary of an unknown word**
    - In English, words (spacing units) which are not found in a dictionary are unknown words
    - In Korean, only subparts of them or themselves are unknown morphemes

Deep Learning Applied NLP Applications

Deep Learning Applied NLP Applications : Index

- Named Entity Recognition
- Image Caption Generator
- Dialogue System Model
- Sentiment Analysis
- Google Neural Machine Translation

Named Entity Recognition

For demonstration, a Named Entity Recognition model developed by the NLP&AI lab is presented

→ An example of Deep Learning applied to Word embedding.

Features of the presented Demo System

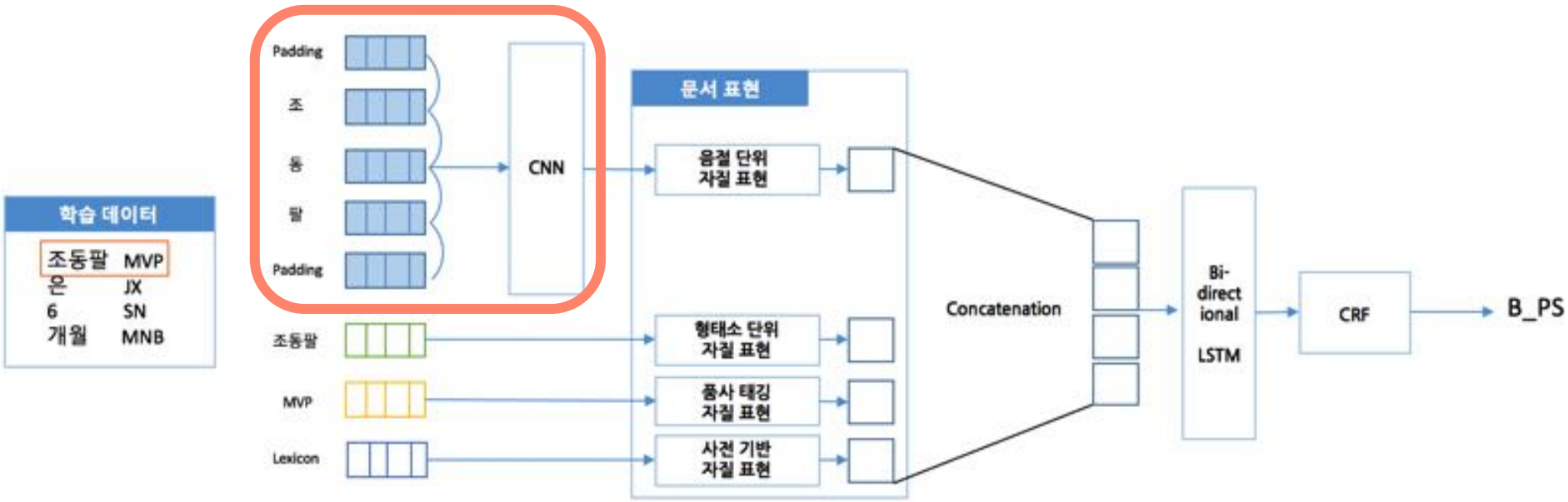
- Feature Representation of Korean syllable units by using CNN
- Feature Representation of morpheme unit by using GloVe vector
- Feature Representation using part-of-speech tagging and lexicons
- Constructed Feature training using bi-directional LSTM and CRF

- By presenting State-of-the-Arts performances in the 2017 국어 정보처리 시스템 Contest, it has won the Golden awards.

Named Entity Recognition

Feature Representation of Korean syllable units by using CNN

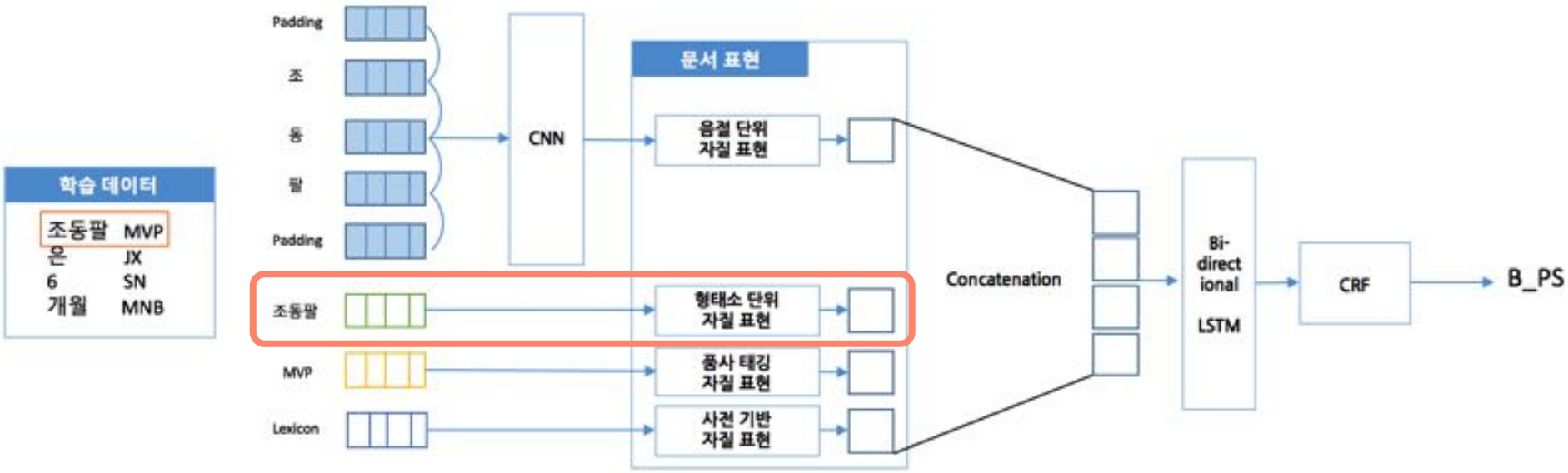
- CNN was used to extract features based on character embeddings



Named Entity Recognition

Feature Representation of morpheme unit by using GloVe vector

- In order to represent the morpheme unit's features, an embedding space using the GloVe vector was constructed
- In order to train the GloVe vector, approximately 3.45 million Wikipedia data consisted of Korean were used



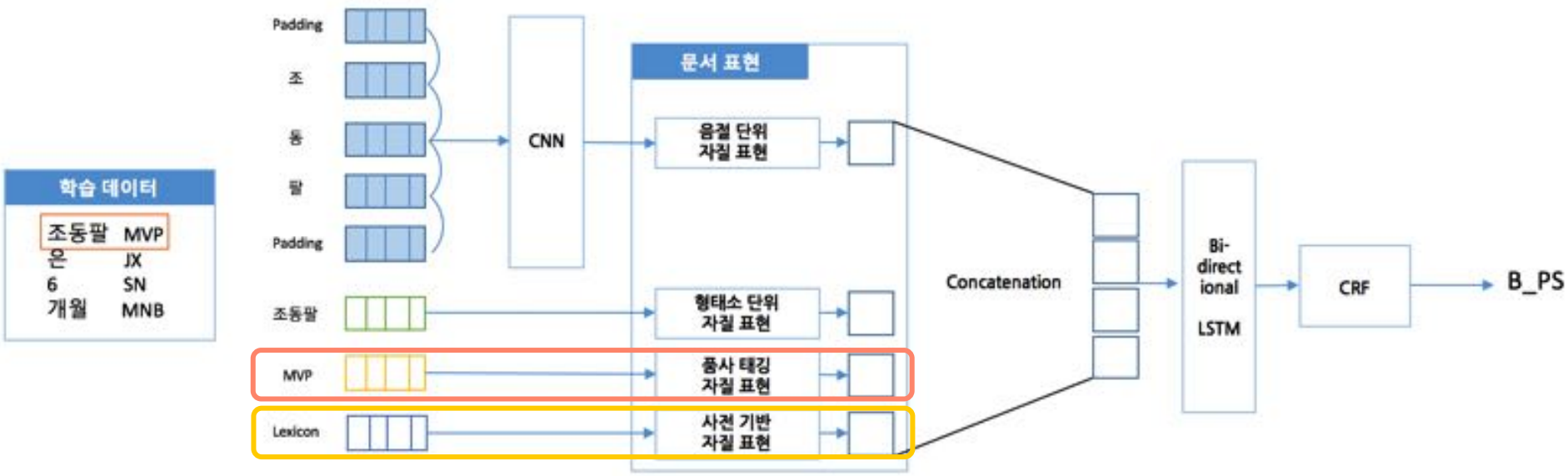
Named Entity Recognition

Feature representation using part-of-speech tagging information

- In the learning data, there were part-of-speech tagging information about the words that can be used when considering the association with the preceding and succeeding words which was utilized for feature representation

Representing Features by using lexicon information

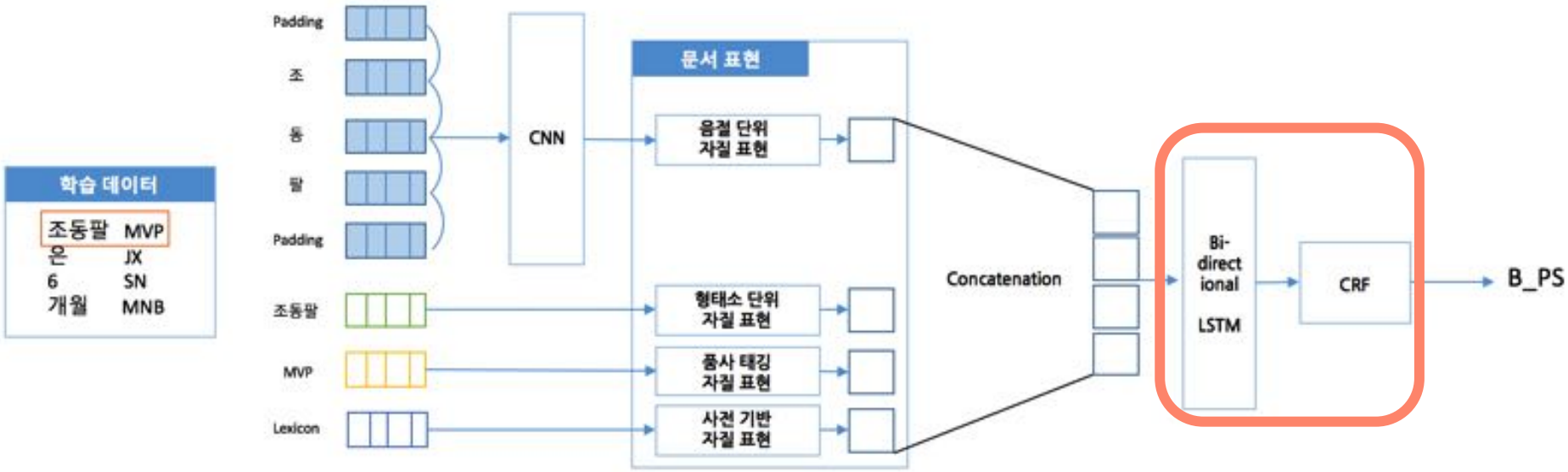
- In order to represent lexicon based features, the gazette lexicon was used



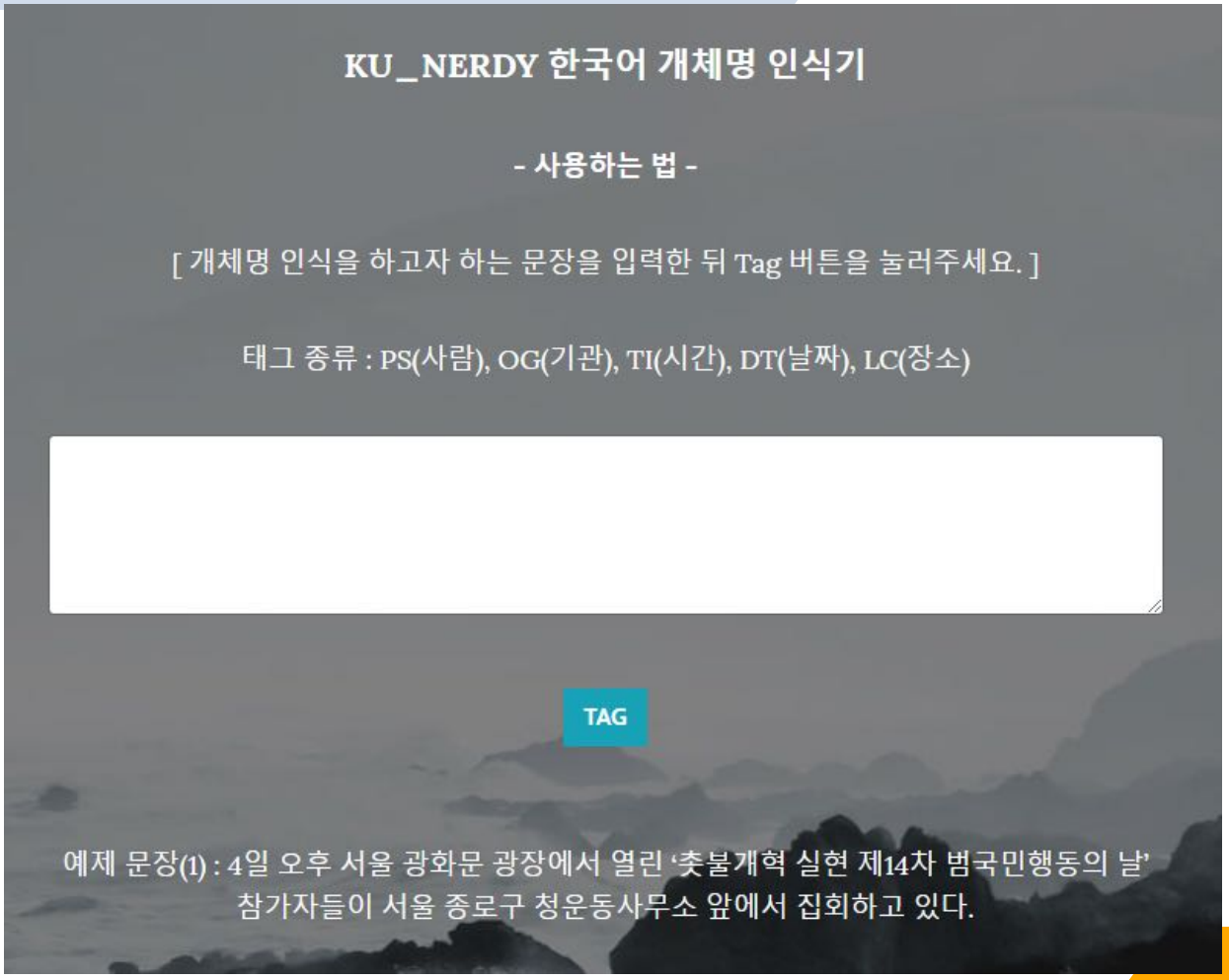
Named Entity Recognition

Constructed Feature training using bi-directional LSTM and CRF

- Calculate the hidden state of each morpheme's information by using the constructed document representation as an input to the bi-directional LSTM
- The bi-directional LSTM computes the hidden state by expressing more rich sentence information by considering the preceding and succeeding sentence information for each sentence.
- The CRF calculates the conditional probability for the hidden state calculated in the previous step and predicts the object name corresponding to the given morpheme



Demo: Named Entity Recognition



NER 실습

Image Caption Generation

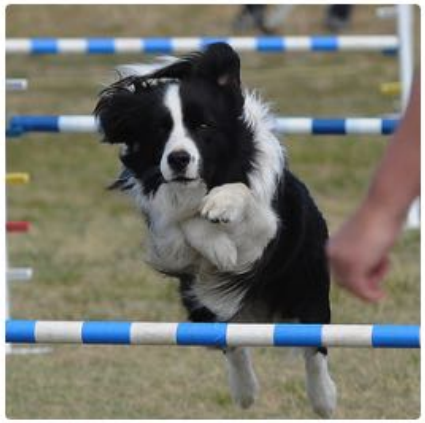
Caption Generation

- Given a digital image, a model that generates a textual description of the contents of the image.

Consists of 2 parts

- Analyze image and transfer the processed image information
- Language models are used to create the image's caption

"girl in pink dress is jumping in air."



"black and white dog jumps over bar."

"young girl in pink shirt is swinging on swing."

Image Caption Generation

For Demonstration, an Image Caption Generator model used in the NLP&AI lab is presented.

→ This model is an example of utilizing deep learning applied machine translation and of image analysis through CNN

The model proceeds as followed

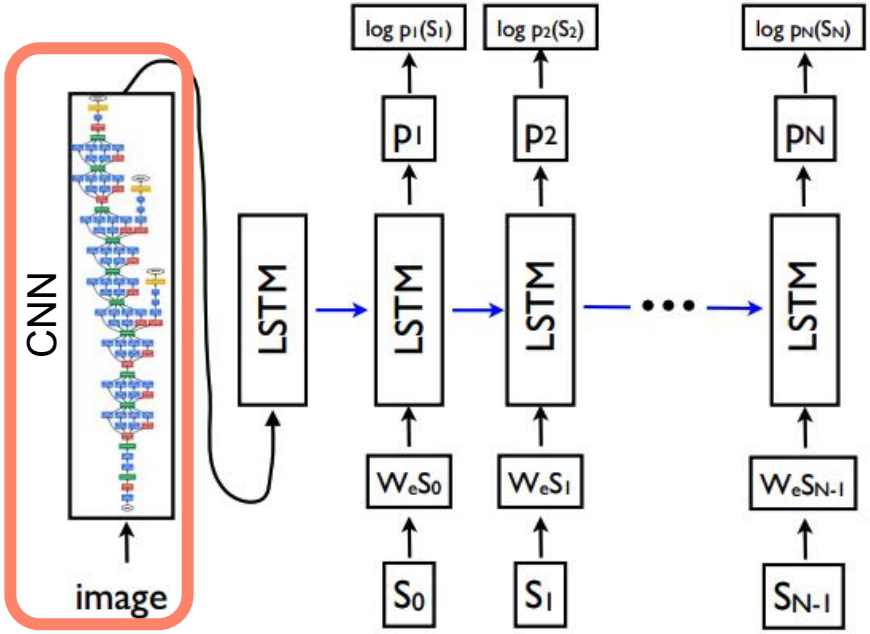
1. Features are extracted from the input image through object detection
2. With the extracted features and the image is analyzed through the CNN generating a vector space representing the image
3. By utilizing a pre-trained data driven MT model, the MT model translates the generated vector space treating it as a unknown 'foreign language'
4. The MT model generates a caption describing the input image

This model has been previously trained with the MSCOCO 2014 dataset to enhance the accuracy of its caption generation

Image Caption Generation

Analyze image through CNN encoder

- For image classification, the machine translator model's encoder RNN has been replaced with a pre-trained CNN
- The CNN embeds the input image in to fixed-length vector

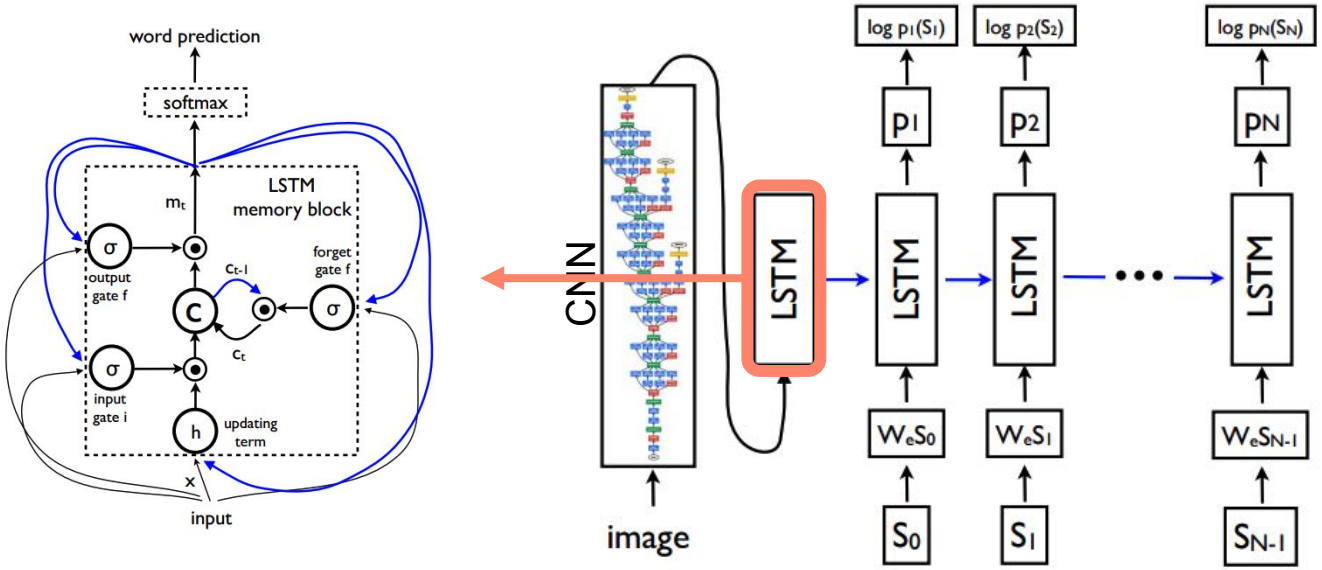


※ N is the length of the sentence

Image Caption Generation

LSTM based Sentence Generator

- The core of the LSTM model is a memory cell c encoding knowledge at every time step of what inputs have been observed up to its current step.
- The LSTM model is trained to predict each word of the sentence after it has seen the image as well as all preceding words.



※ N is the length of the sentence

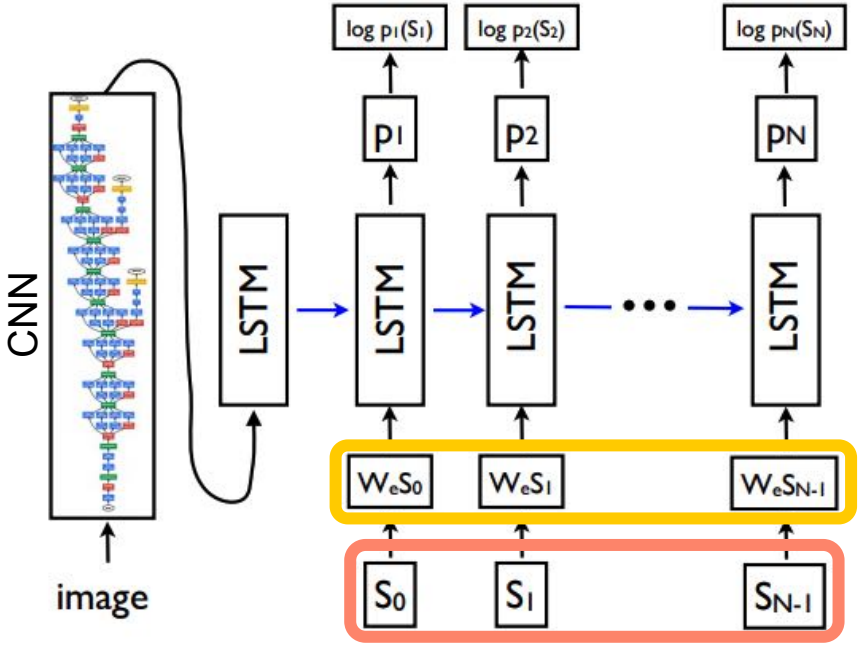
Image Caption Generation

$\{S\downarrow 0 , S\downarrow 1 , S\downarrow 2 , \dots , S\downarrow N-1 \}$

- S refers to the words of the caption by transcription

$\{W\downarrow e\, S\downarrow 0 , W\downarrow e\, S\downarrow 1 , W\downarrow e\, S\downarrow 2 , \dots , W\downarrow e\, S\downarrow N-1 \}$

- A word embedding vector corresponding to each transcription



※ N is the length of the sentence

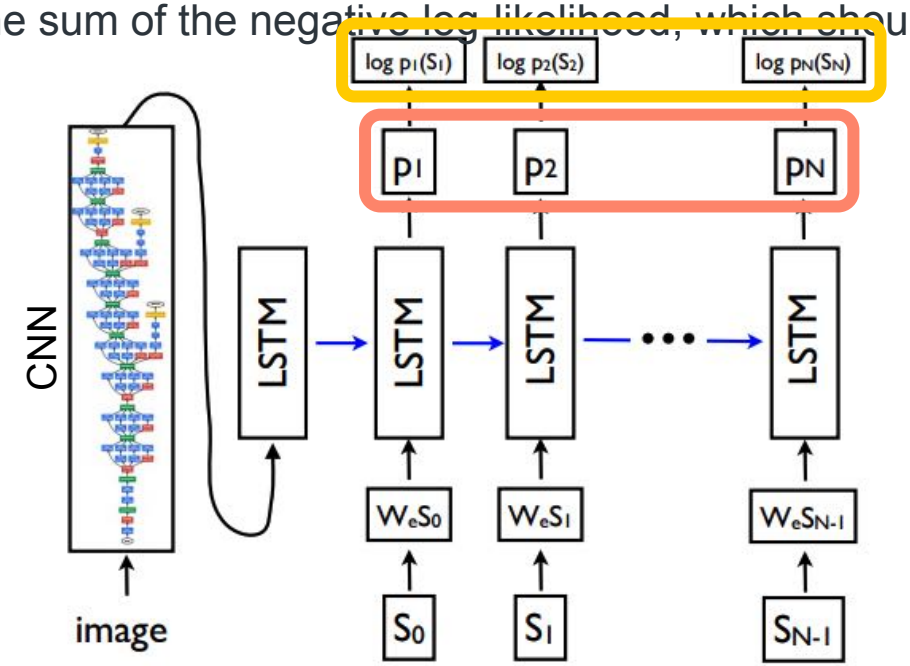
Image Caption Generation

$$\{p_1, p_2, p_3, \ldots, p_N\}$$

- Probability distribution generated by the model for the next word of the sentence

$$\{\log p_1(S_1), \log p_2(S_2), \ldots, \log p_N(S_N)\}$$

- Log-likelihood of the correct word at each step
- The loss is the sum of the negative log-likelihood, which should be minimized



※ N is the length of the sentence

Image Caption Generation

Finally by emitting a pre-defined stop word, the LSTM signals that a complete sentence has been generated which concludes the image caption generation.

This model was appraised for its performance in the MSCOCO challenge of 2015 being second only to an actual person.
Yet it is still highly dependent on the contents of its training data and exhibits a sharp drop in performances when shown an object that goes beyond its training.

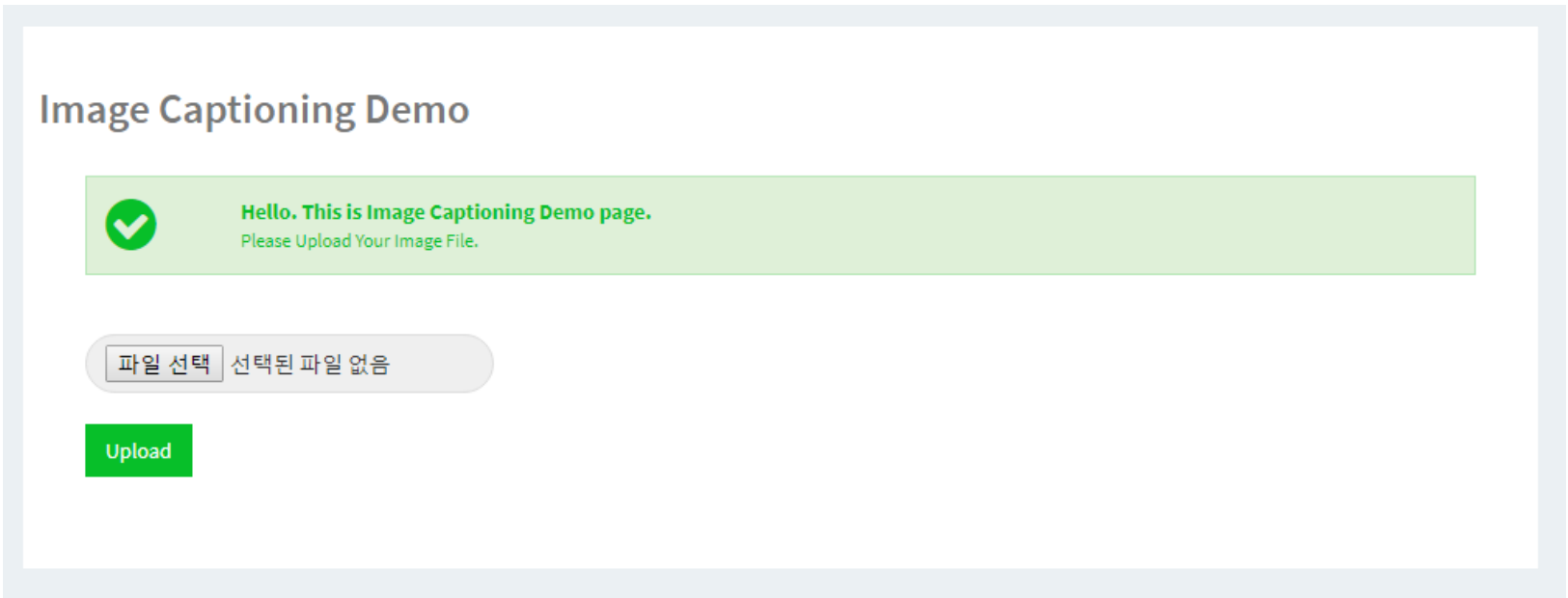
Captioning Leaderboard

Table-C5 Table-C40 **Challenge2015**

Copy to Clipboard | Export to CSV | Search:

	M1	M2	M3	M4	M5	date
Human	0.638	0.675	4.836	3.428	0.352	2015-03-23
Google	0.273	0.317	4.107	2.742	0.233	2015-05-29
MSR	0.268	0.322	4.137	2.662	0.234	2015-04-08
Montreal/Toronto	0.262	0.272	3.932	2.832	0.197	2015-05-14
MSR Captivator	0.250	0.301	4.149	2.565	0.233	2015-05-28
Berkeley LRCN	0.246	0.268	3.924	2.786	0.204	2015-04-25

Demo: Image Caption Generation



Dialogue System

The following goal-oriented Dialogue system model was developed at the NLP&AI lab

→ This model is an example of how deep learning can be applied to dialogue systems and enhance its performance.

Features of the presented model:

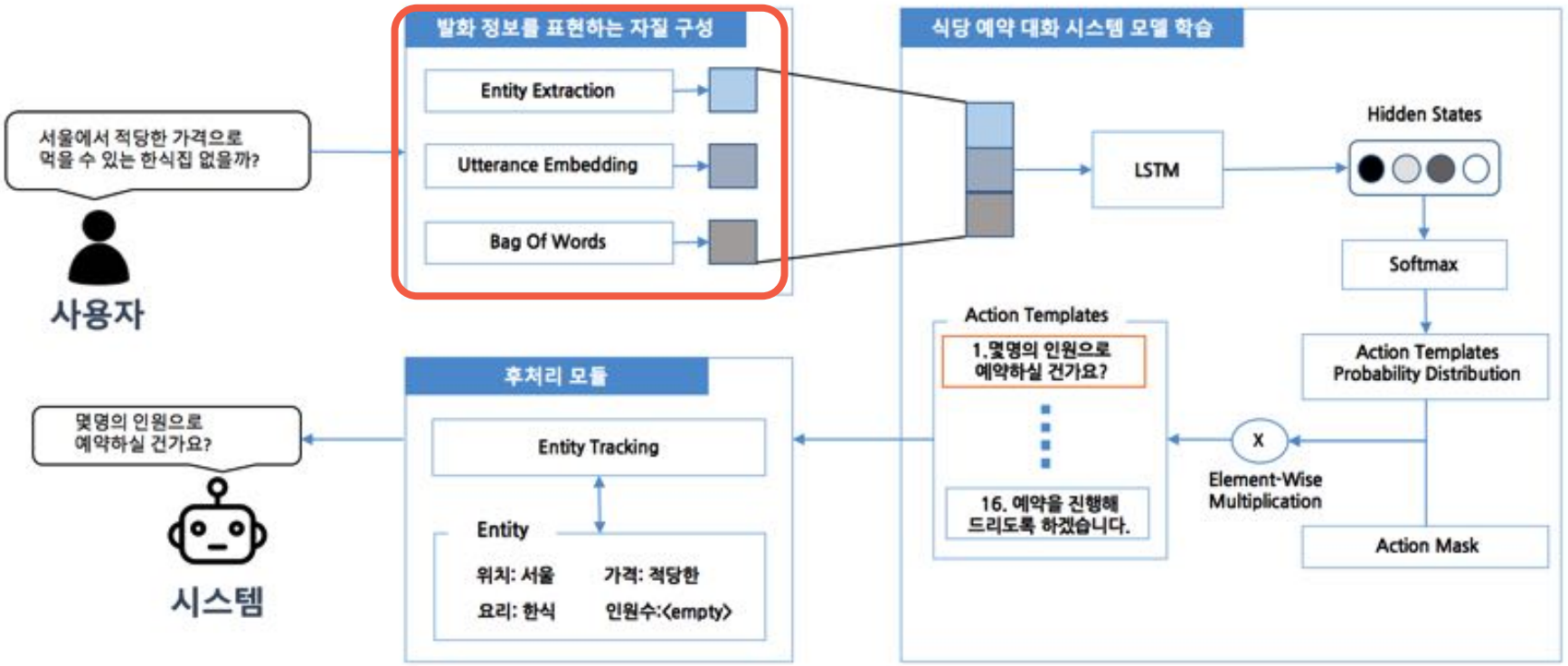
1. Based on a RNN(Recurrent Neural Network) model
2. Conversation system is built by learning conversation data by end-to-end learning method.
3. To overcome the drawbacks of end-to-end learning's great need of data, a hybrid code network structure that combines domain-specific knowledge and enables a relatively smaller training dataset to be possible.
4. The models dialogue is aimed at reserving restaurants.

759 learning conversation data and 190 test conversation data were used to study and train this model.

Dialogue System for Restaurant Reservation

Spoken Information consists of 3 features

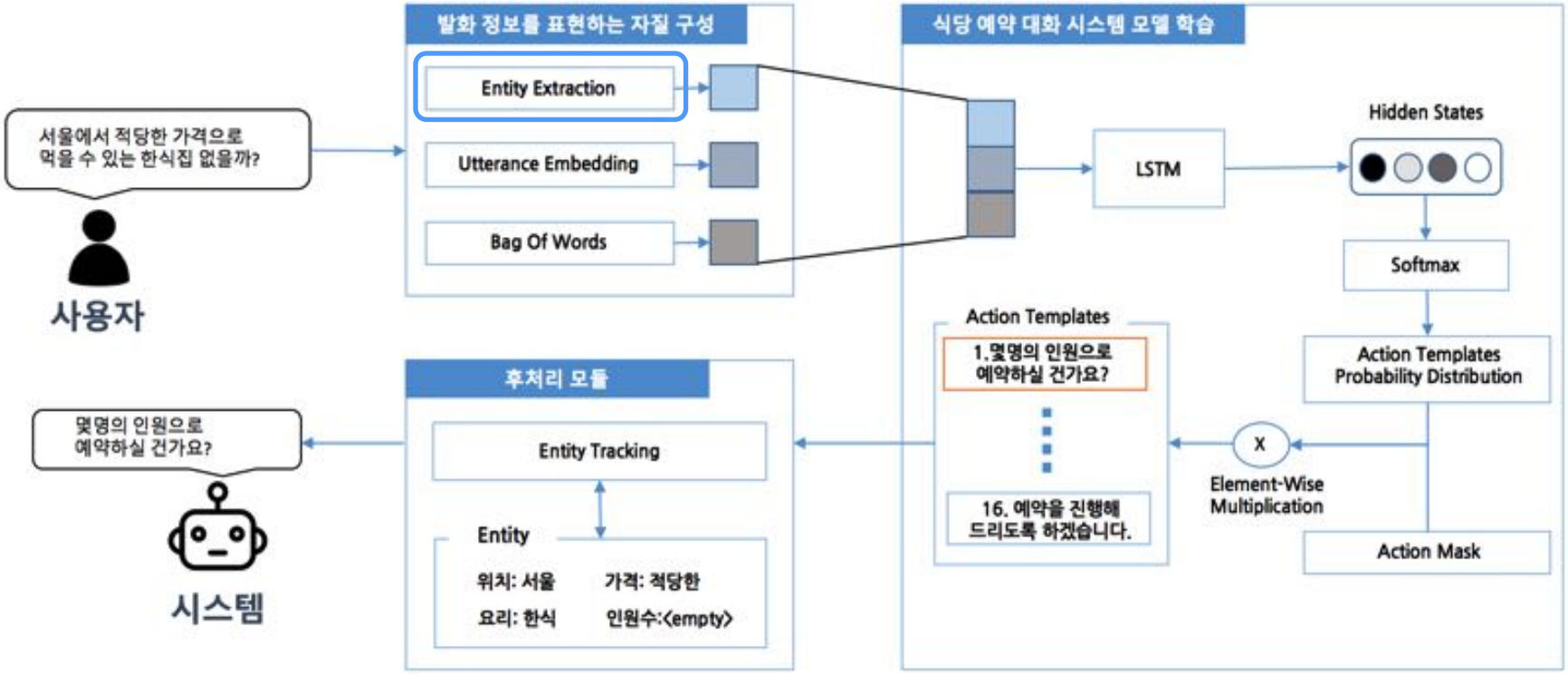
- Entity Extraction
- Utterance Embedding
- Bag of Words



Dialogue System for Restaurant Reservation

Entity Extraction

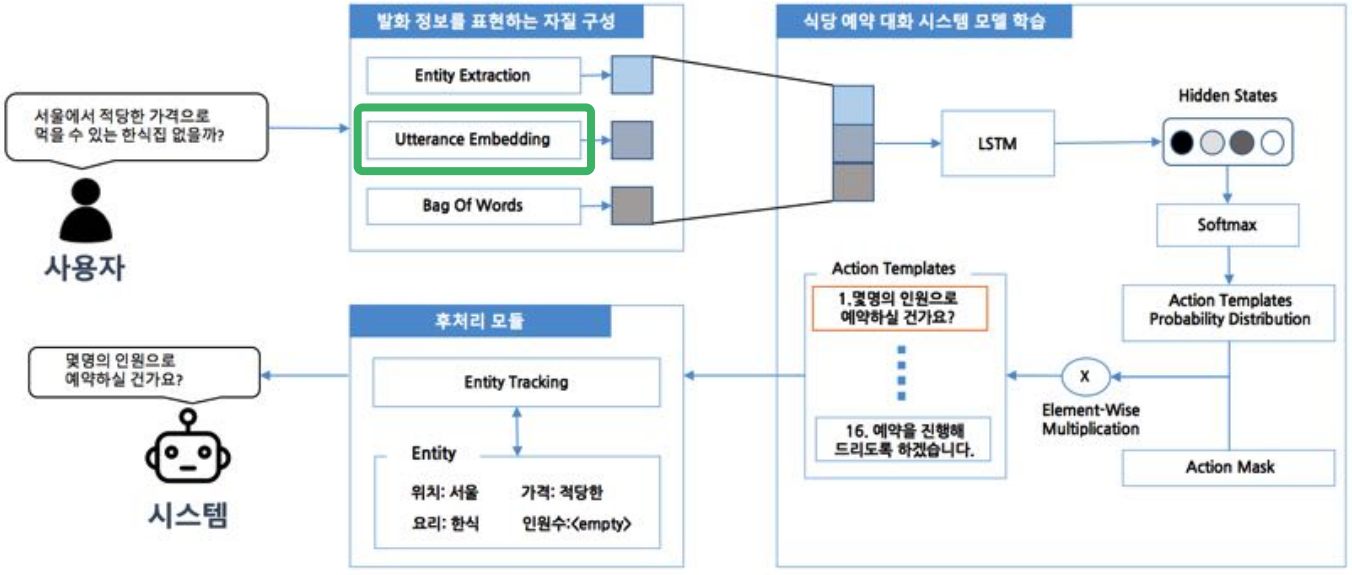
- Attributes needed for the oriented goal (reserving restaurants) are defined as entities such as location of the restaurant, price of the food, type of the food, the number of people and etc.



Dialogue System for Restaurant Reservation

Utterance Embedding

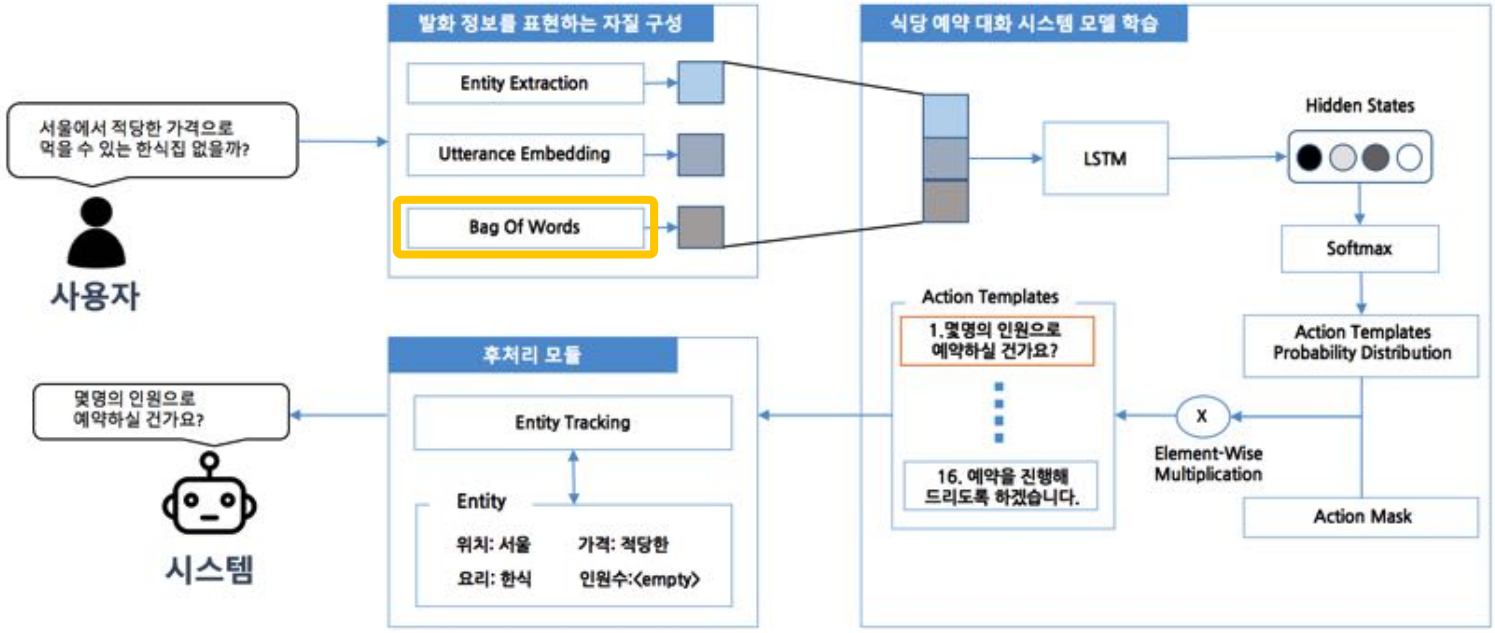
- The embedding module embeds the user's utterance using the word2vec model in order to reflect the semantics characteristic from the utterance of the user, and then composes it as the qualities.
- While N is the total number of words and the i-th word of the document is expressed as a vector value in the word embedding space as v (i), $1/N \sum i=1 \uparrow N \square v(i)$ is the vector mean of each word constituting the utterance



Dialogue System for Restaurant Reservation

Bag of Words

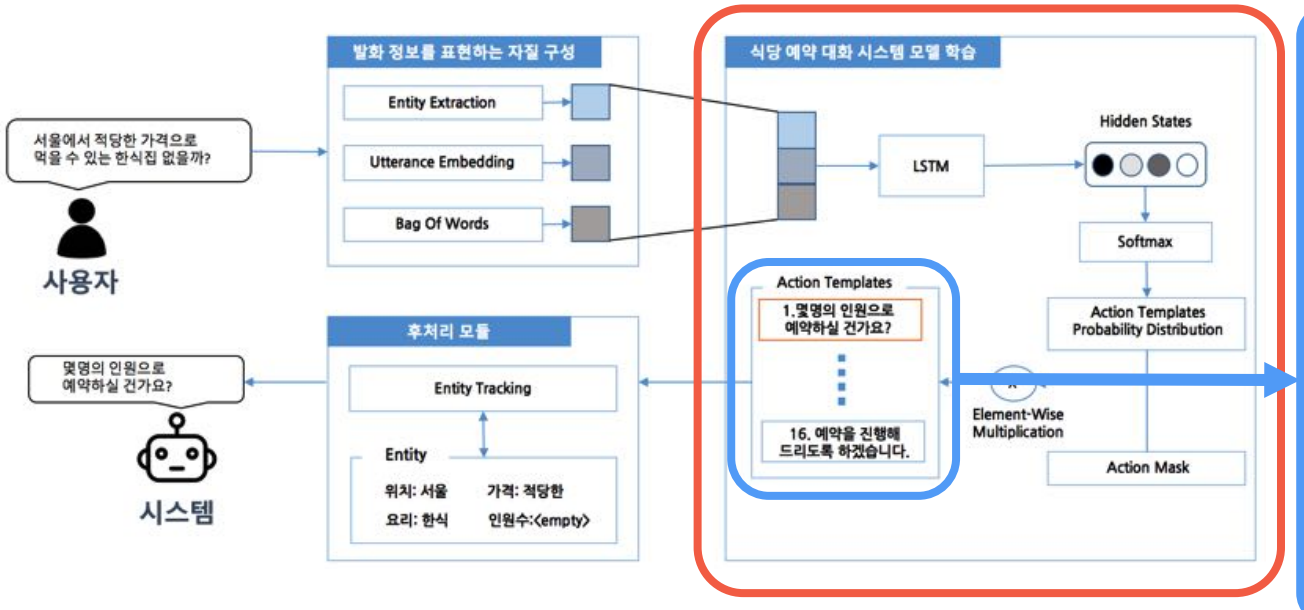
- To construct the user's utterance into a bag of words, a dictionary is formed based on a set of words from the training dataset of conversations.
- The bag of word's features expressing the user's utterance was constructed by marking the appearances of words that constitute each utterance.



Dialogue System for Restaurant Reservation

Training the LSTM based dialogue system

- To reflect domain specific knowledge in the system response, we define an action template as shown below.
- The features from the speech information are used as inputs, and the LSTM calculates the hidden state using the input.



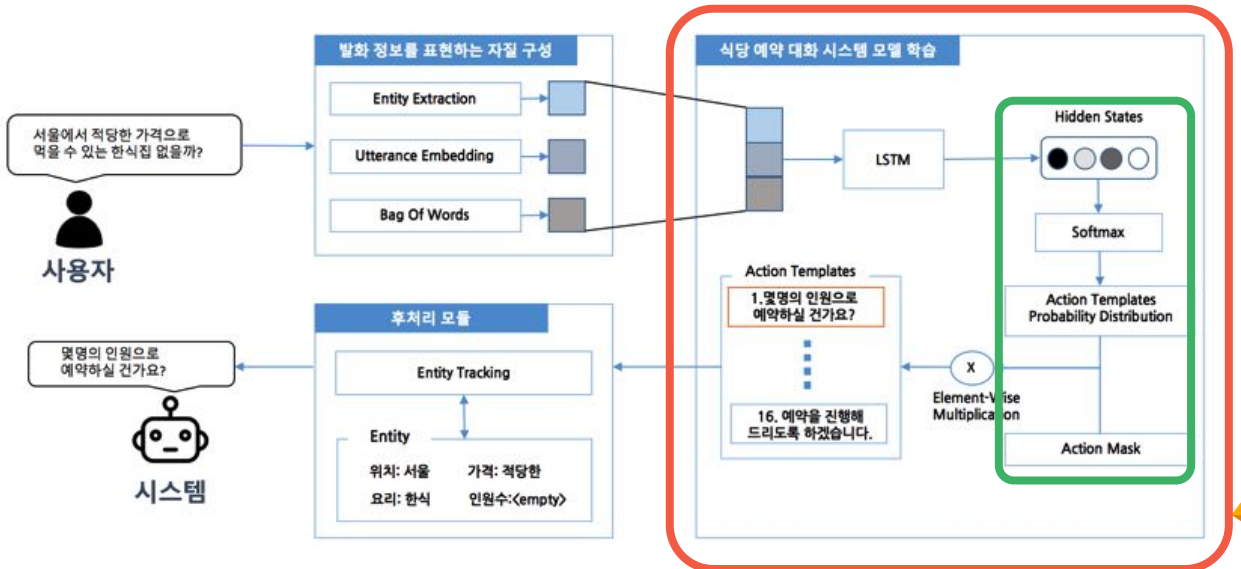
Action Templates

0. api_call <음식종류> <위치> <인원수> <가격>
1. 가격의 범위는 어느정도로 생각하세요
2. 감사합니다
3. 네 또 변경하실게 있나요
4. 다른 리스트를 보여드릴게요
5. 또 도와드릴게 있나요
6. 몇명의 인원으로 예약하실 건가요
7. 안녕하세요 어떻게 도와드릴까요
8. 알겠습니다
9. 어떤 종류의 요리를 좋아하나요
10. 예약을 진행해드리도록 하겠습니다
11. 위치는 <info_address> 입니다
12. 위치는 어디에 있어야 하나요
13. 이 리스트는 어떤가요: <restaurant>
14. 전화번호는 <info_phone> 입니다
15. 좋아요 몇 가지 리스트를 보여드릴게요

Dialogue System for Restaurant Reservation

Training the LSTM based dialogue system

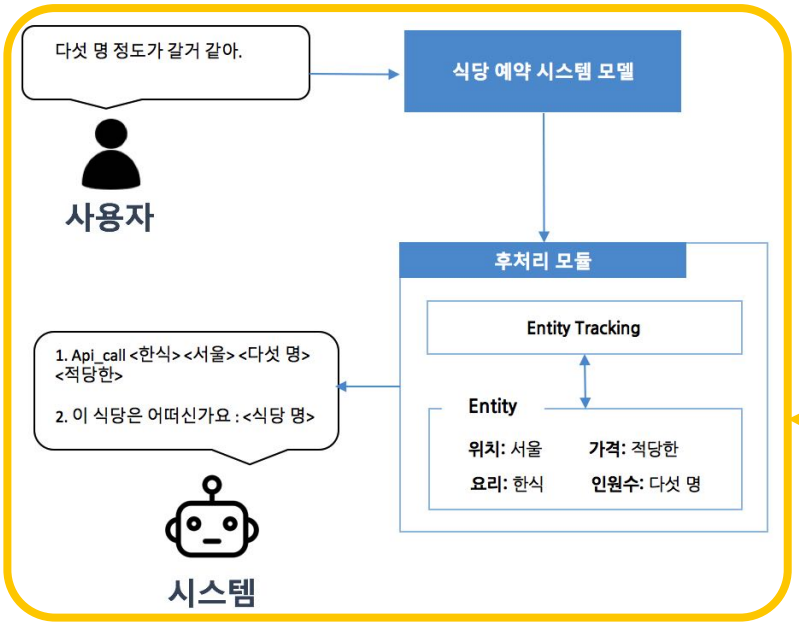
- With the hidden state as input, the softmax calculates the probability distribution value for each action template.
- Each probability distribution value is then normalized through the product of the action mask and the components.
- Finally The model's training proceeds trying to minimize the cross-entropy value between the normalized values and the labeled correct system response from the actual learning data set.

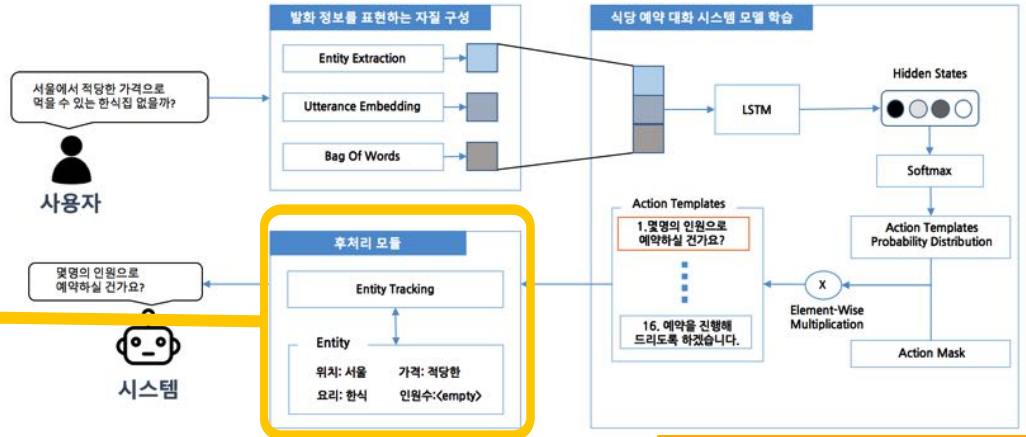


Dialogue System for Restaurant Reservation

Post Processing Module

- Determine whether the entity is needed in the utterance of the selected action template through the dialogue system model predicted results for user utterance.
- If needed, The objects extracted from the previous speech are traced so that the entities can be responded together.





Sentiment Analysis

The following Sentiment Analysis model is a demo developed from Stanford University

→ This model is an introduced the application of Deep learning to Sentiment Analysis and greatly improved it's performance.

Features of the presented model:

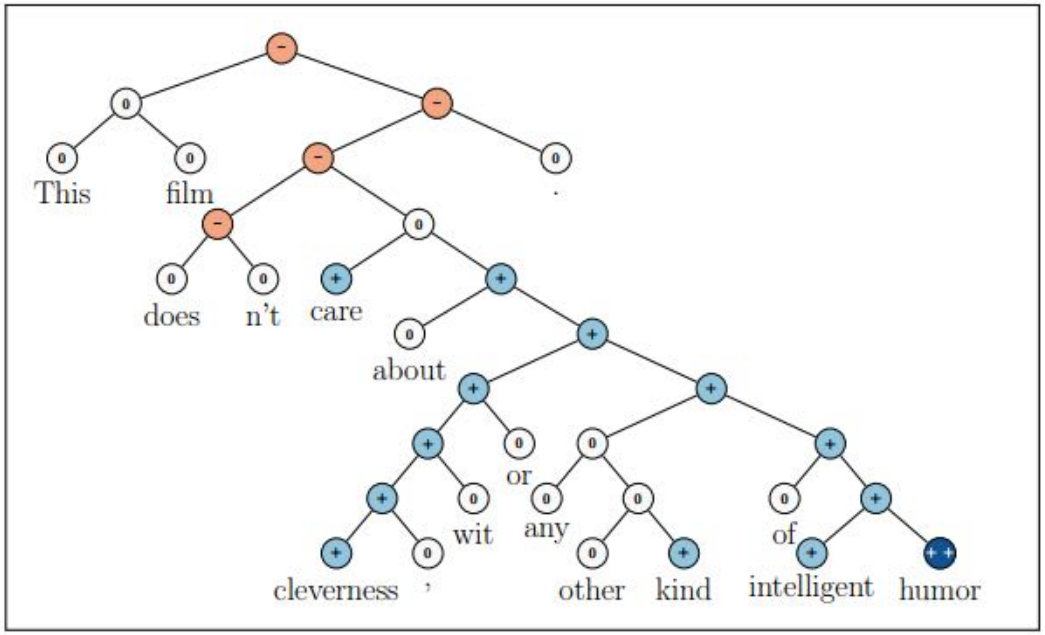
1. A new type of RNN that is built upon grammatical structures.
2. Introduces a dataset called the Sentiment Treebank and present new challenges for sentiment compositionality.
3. By building up a representation of whole sentences based on the sentence structure, it computes the sentiment based on how words compose the meaning of longer phrases, unlike previous models just looking at words in isolation.

This model was presented in 2013, and has pushed the state of the art boundary in single sentence positive/negative classification from 80% to 85.4%.

Sentiment Analysis

The Stanford Sentiment Treebank is the first corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language.

→ This new dataset allows the analysis of sentiment intricacies and the capture of complex linguistic phenomena as shown below.



Sentiment Analysis

Sentiment Analysis | Information | **Live Demo** | Sentiment Treebank | Help the Model | Source Code

Please enter text to see its parses and sentiment prediction results:

```
This movie doesn't care about cleverness, wit or any other kind of intelligent humor.
Those who find ugly meanings in beautiful things are corrupt without being charming.
There are slow and repetitive parts, but it has just enough spice to keep it interesting.
```

You can also upload a file (limit 200 lines): 파일 선택 선택된 파일 없음 Show trees in binary form Submit

Sentiment Analysis

Stanford Sentiment Treebank

- Based on the corpus of movie reviews from the *rottentomatoes.com* website.
- Each label is extracted from a longer movie review and reflects the writer's overall intention for this review.
- The Stanford Parser (Klein and Manning,2003) splits the snippet into multiple sentences and the Amazon Mechanical Turk is used to label the resulting 215,154 phrases.

Sentiment Analysis

To capture the compositional effects from the dataset with higher accuracy, a new model based on RNN was introduced.
→ Recursive Neural Tensor Network(RNTN)

Recursive Neural Tensor Network (RNTN)

- Motivated idea:
  → Can a single, more powerful composition function perform better and compose aggregate meaning from smaller constituents more accurately than many input specific ones?
- Main idea is to use the same, tensor-based composition function for all nodes.

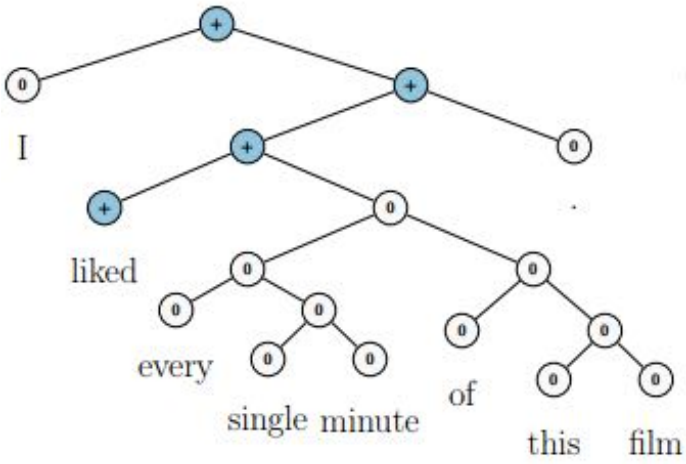
Features of RNTN

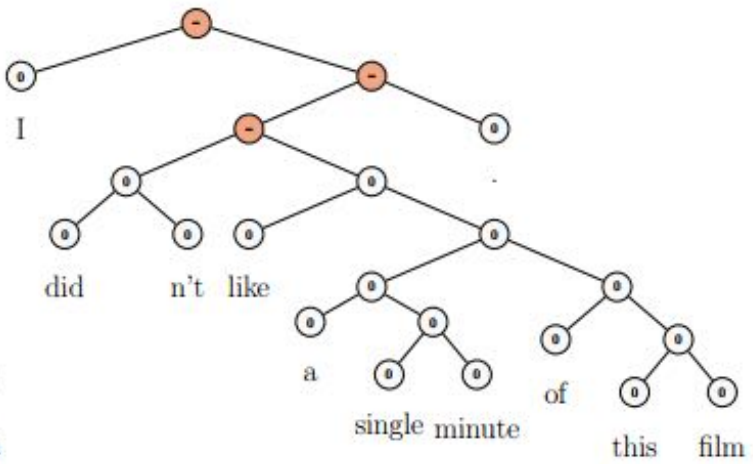
- Take phrases of any length as input phrases.
- Represent a phrase through word vectors and a parse tree .
- Compute vectors for higher nodes in the tree using the same tensor-based composition function.

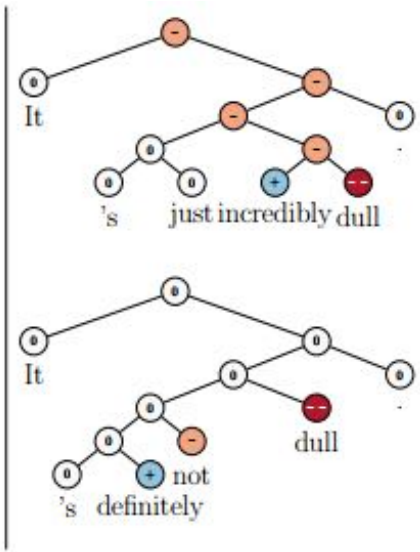
Sentiment Analysis

Result of Sentiment Analysis

- Computes the sentiment based on how words compose the meaning of longer phrases as shown below
- The following method has shown an accuracy in positive/negative classification from 80% to 85.4% compared to existing methods.







Google Neural Machine Translation System

A Deep Learning applied Machine Translation model developed by Google

→ The most successfully NLP task that applied deep learning as well as the most commonly used.

Features of the Google NMT model

- Has an end-to-end learning approach.
- Overcome many disadvantages of conventional phase-based MT models
- Relatively simple and perform wells compared to existing models
- Uses 8 LSTM encoders and 8 LSTM decoders
  → LSTM tends to perform better as it is deeper, as it did here

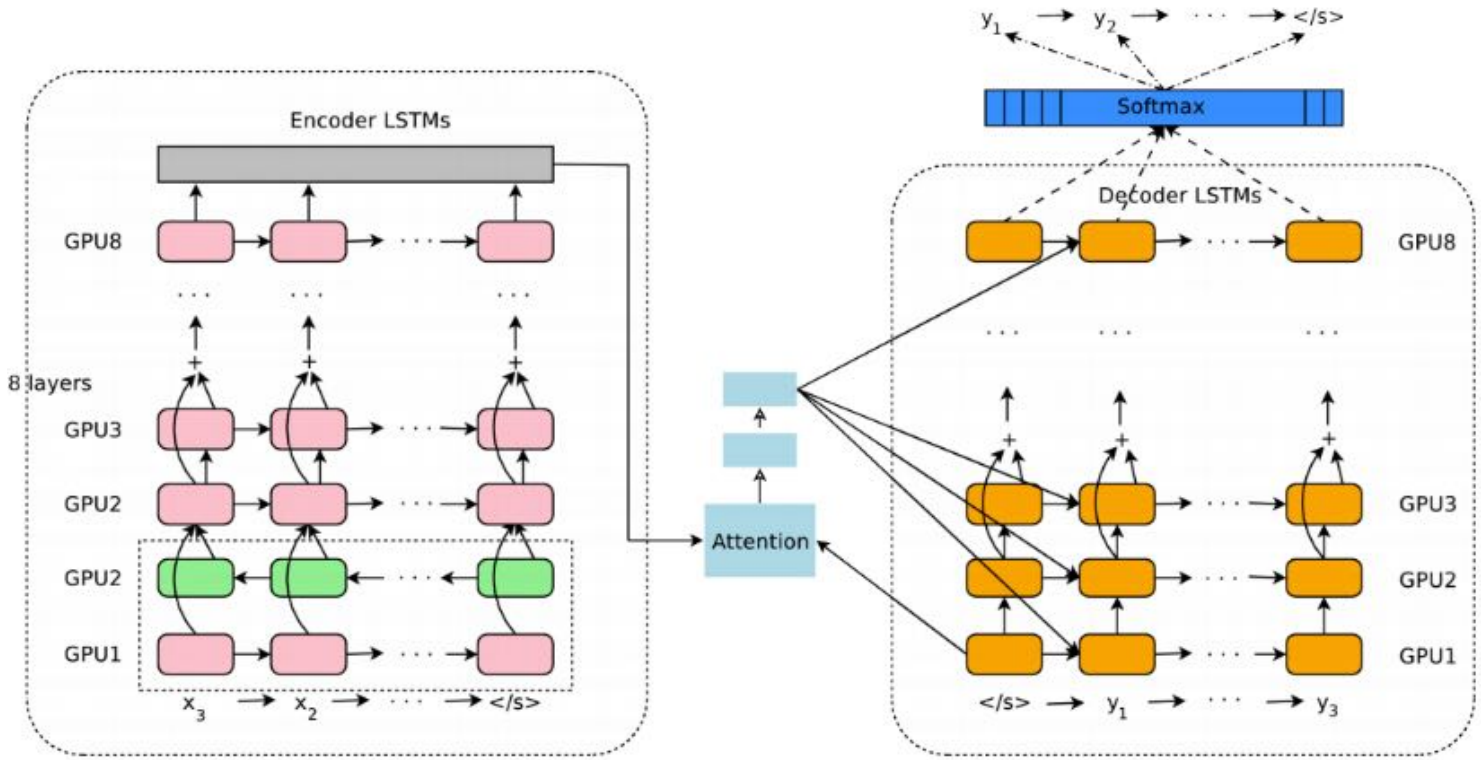
Google NMT's weaknesses

- Requires a lot of computation and computing power
- Requires very huge amounts of data for training
- Weak against Rare words
  → TO over come this, the GNMT used wordpiece

Google Neural Machine Translation System

Model's Structure

- Encoder network that processes input strings (Left)
- Decoder network that processes output strings (Right)
- Attention network (Middle)



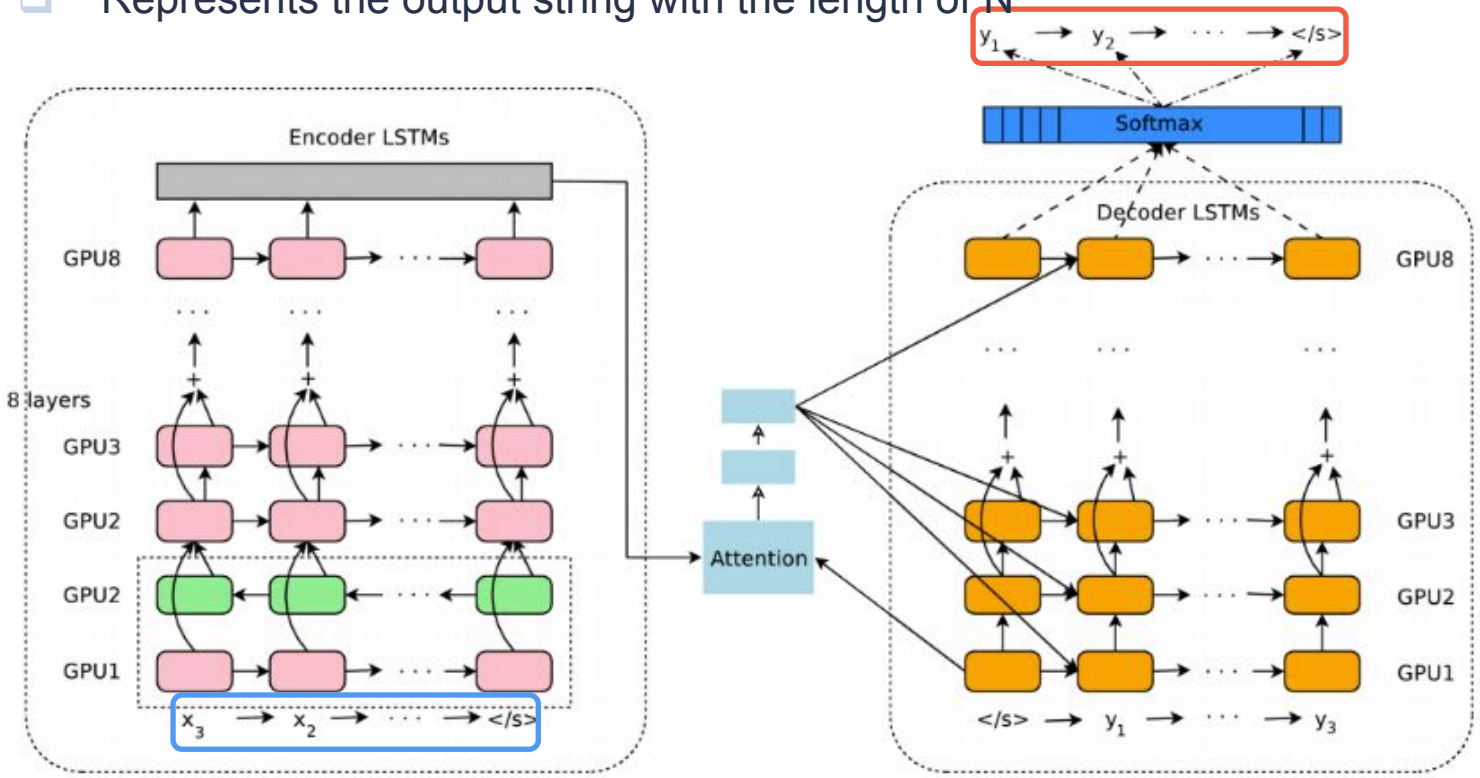
Google Neural Machine Translation System

$X=\{x\downarrow 1 , x\downarrow 2 , \ldots , x\downarrow M\}$

- Represents the input string with the length of M

$Y=\{y\downarrow 1 , y\downarrow 2 , \ldots , y\downarrow N\}$

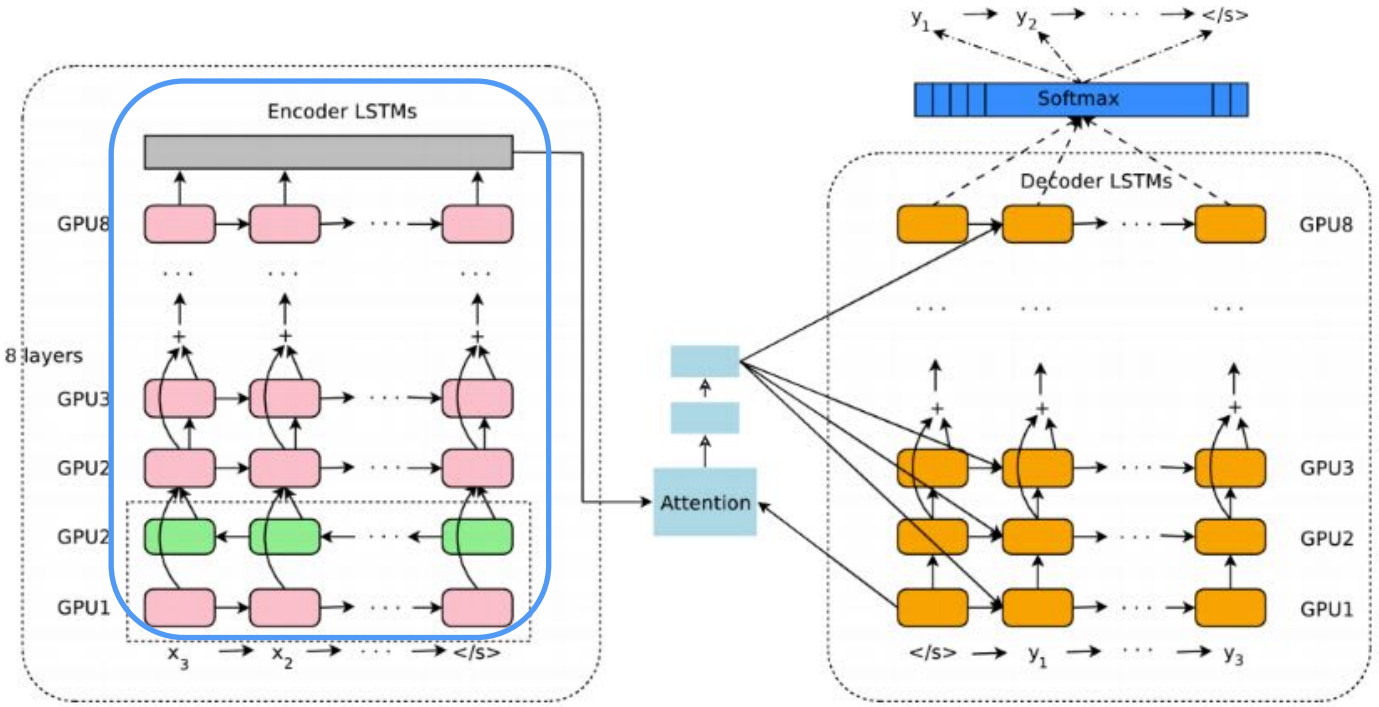
- Represents the output string with the length of N



Google Neural Machine Translation System

Encoder LSTM

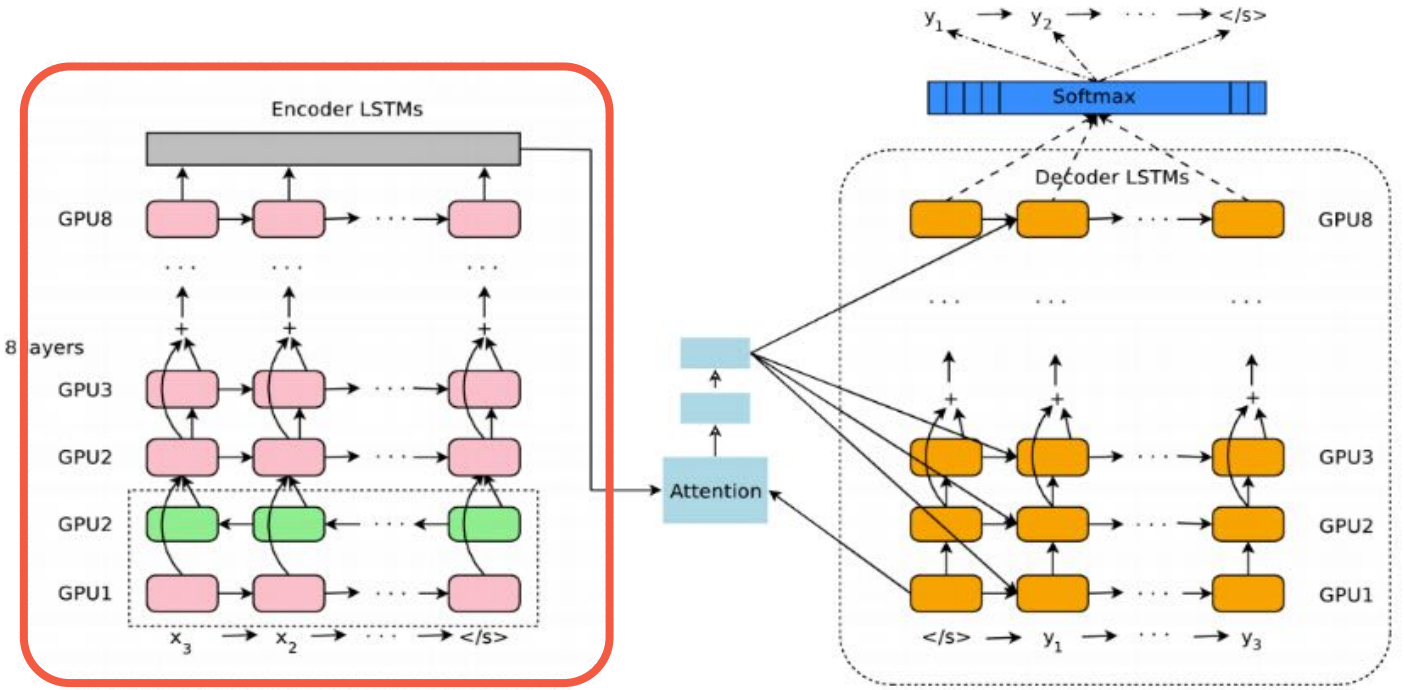
- The encoder LSTM is simply a function of the following form
- $x\downarrow 1 , x\downarrow 2 , \ldots , x\downarrow M = EncoderRNN(x\downarrow 1 , x\downarrow 2 , \ldots , x\downarrow M)$
- $x\downarrow 1 , x\downarrow 2 , \ldots , x\downarrow M$ is a list of fixed size vectors



Google Neural Machine Translation System

Probability of Y when string X is given $P(Y|X)$

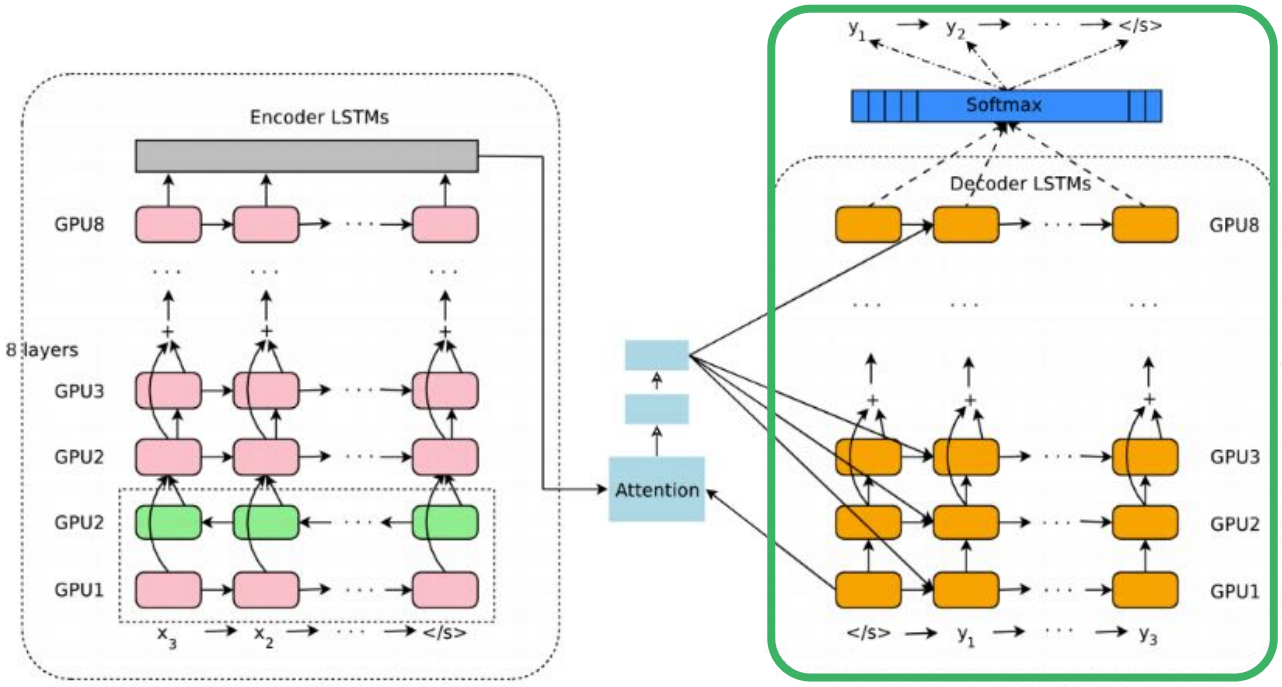
- Can be decomposed as following:
- $PYX = P(Y|\ x\downarrow 1, x\downarrow 2, \ldots, x\downarrow M) = \prod i=1 \uparrow N \square P(y\downarrow i\ |y\downarrow 0,\ y\downarrow 1,\ y\downarrow 2, \ldots, y\downarrow i-1;\ x\downarrow 1, x\downarrow 2, \ldots, x\downarrow M)$
- Note that $y\downarrow 0$ is a special symbol that signals the be of a sentence



Google Neural Machine Translation System

Decoder LSTM

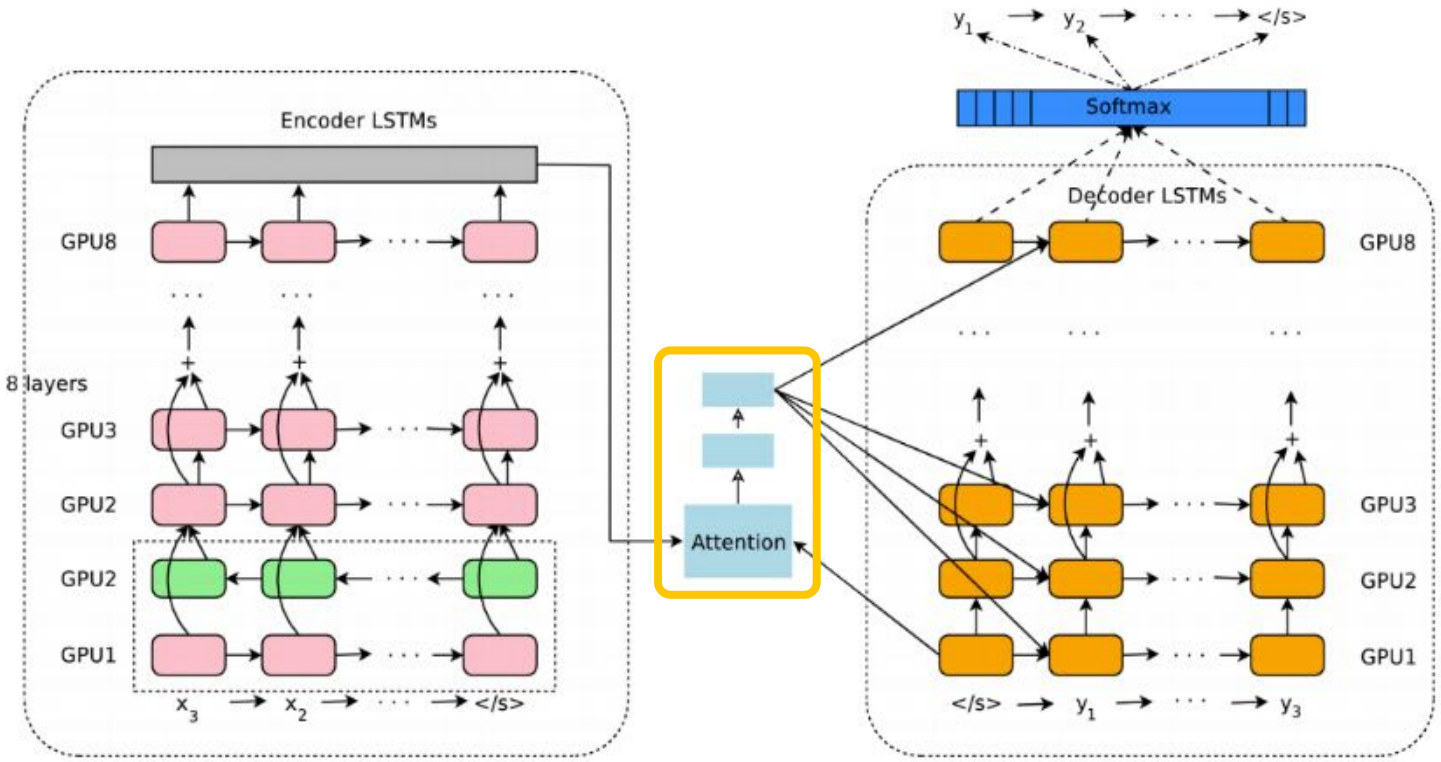
- The decoder is implemented as a combination of an RNN network and a softmax layer.
- Hidden state $y\downarrow 1$ is generated in order to go through the softmax layer to generate a probability distribution over candidate output symbols.



Google Neural Machine Translation System

Attention Model

- Reduce the encoder's burden of having to encode all of the sentence's information into a fixed length vector.



Limitation of Deep Learning

Limitation of AI

Search Facebook

Building Jarvis

MARK ZUCKERBERG · MONDAY, DECEMBER 19, 2016

My personal challenge for 2016 was to build a simple AI to run my home -- like Jarvis in Iron Man. Within 5-10 years we'll have AI systems that are more accurate than people for each of our senses -- vision, hearing, touch, etc, as well as things like language. It's impressive how powerful the state of the art for these tools is becoming.

At the same time, we are still far off from understanding how learning works. Everything I did this year – natural language, face recognition, speech recognition and so on –are all variants of the same fundamental pattern recognition techniques. I spent about 100 hours building Jarvis this year, **but even if I spent 1,000 more hours, I probably wouldn't be able to build a system that could learn completely new skills on its own -- unless I made some fundamental breakthrough in the state of AI along the way**.

In a way, AI is both closer and farther off than we imagine. AI is closer to being able to do more powerful things than most people expect -- driving cars, curing diseases, ... Those will each have a great impact on the world, but **we're still figuring out what real intelligence is.**

What else do we need?

- Intelligence is not just about pattern recognition.
- It is about modeling the world…
  - Explaining and understanding what we see.
  - Imagining things we could see but haven't yet.
  - Problem solving and planning actions to make these things real.
  - Building new models as we learn more about the world.

To read more: Lake, Ullman, Tenenbaum & Gershman, "Building machines that learn And think like people", on arXiv and in press at Behavioral and Brain Sciences.

Bridging the gap

- We are decades away from AI that can build models of the world as flexibly and as deeply as human do, and that is mature enough to deploy at Google of Facebook. But we are starting to understand some of the fundamental principles.
- **Zero-shot learning** : How can we solve a task despite not having received any training examples of that task?

(No training example for the target category)

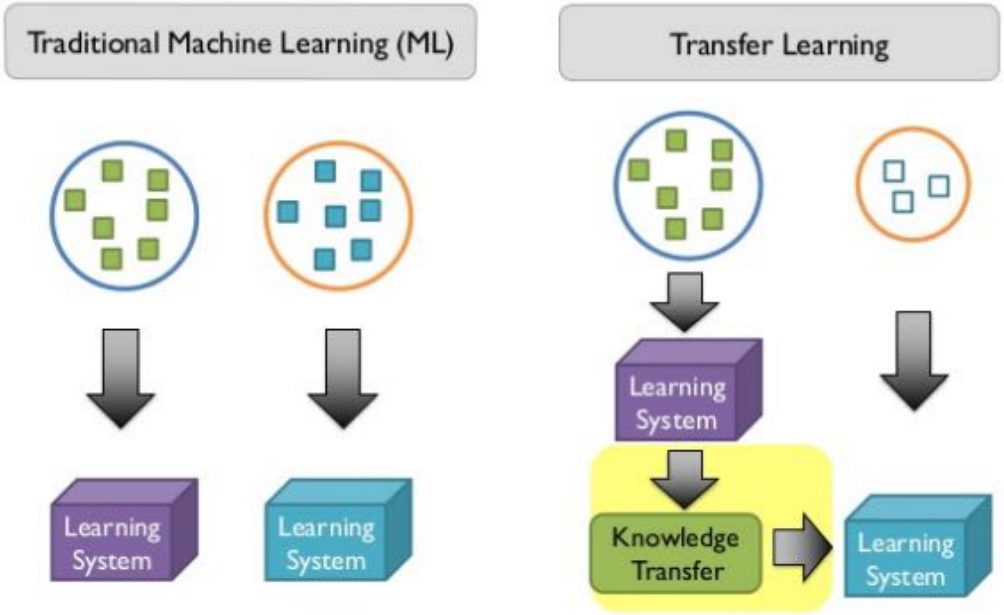
- **One-shot learning** : How can we learn such rich concepts from so little experience – often just a single example?

(Only one training example per category)

- **Commonsense scene understanding**:
- How can we see a whole world of physical objects, their interactions and our own possibilities to act and interact with others – not simply classify patterns in pixels?

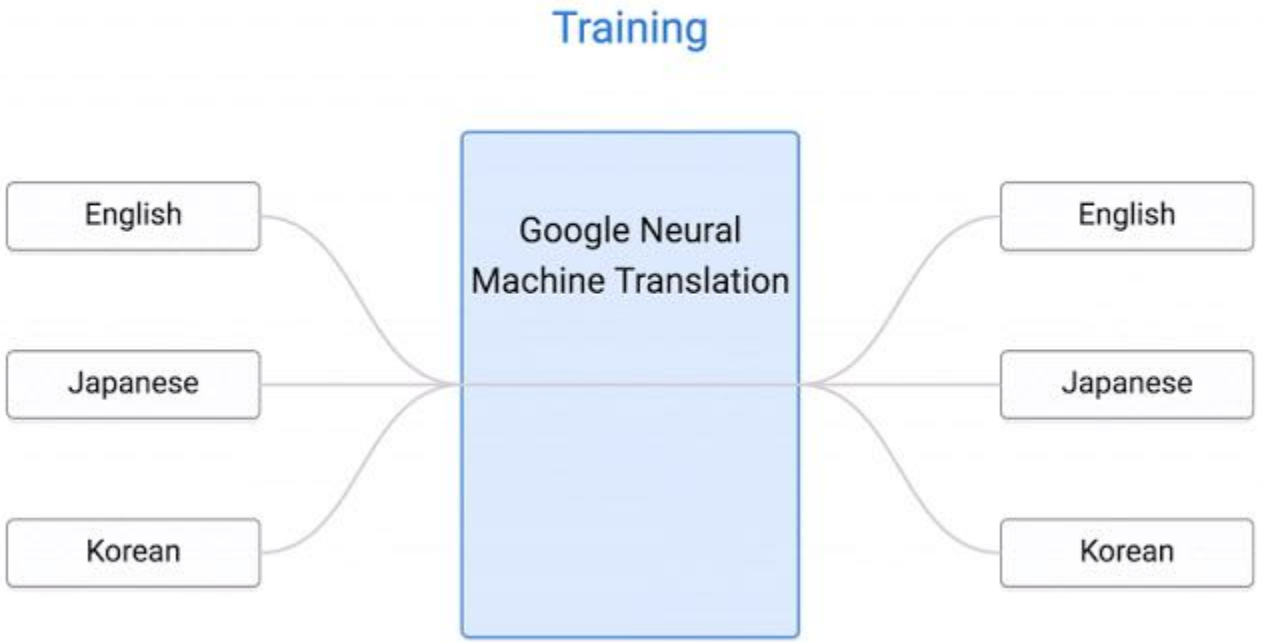
Transfer learning

- Improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.
  - One-shot learning
  - Zero-shot learning



Zero-shot learning

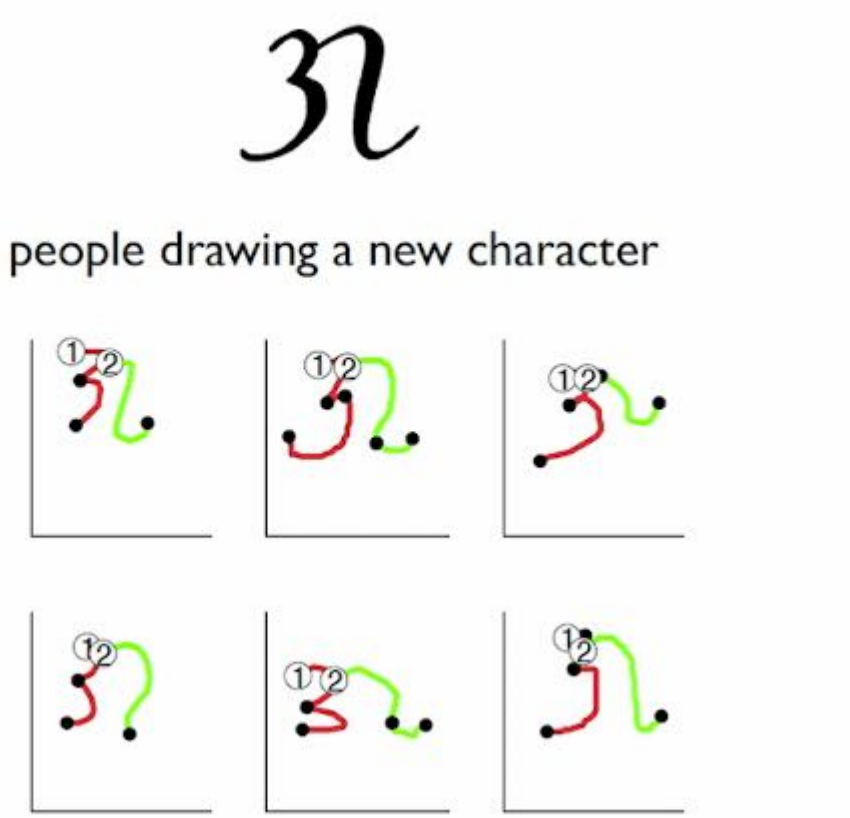
- Google's multilingual neural translation system : Enabling zero-shot translation
- Translation between language pairs never seen explicitly by the system.



One-shot learning (1/3)

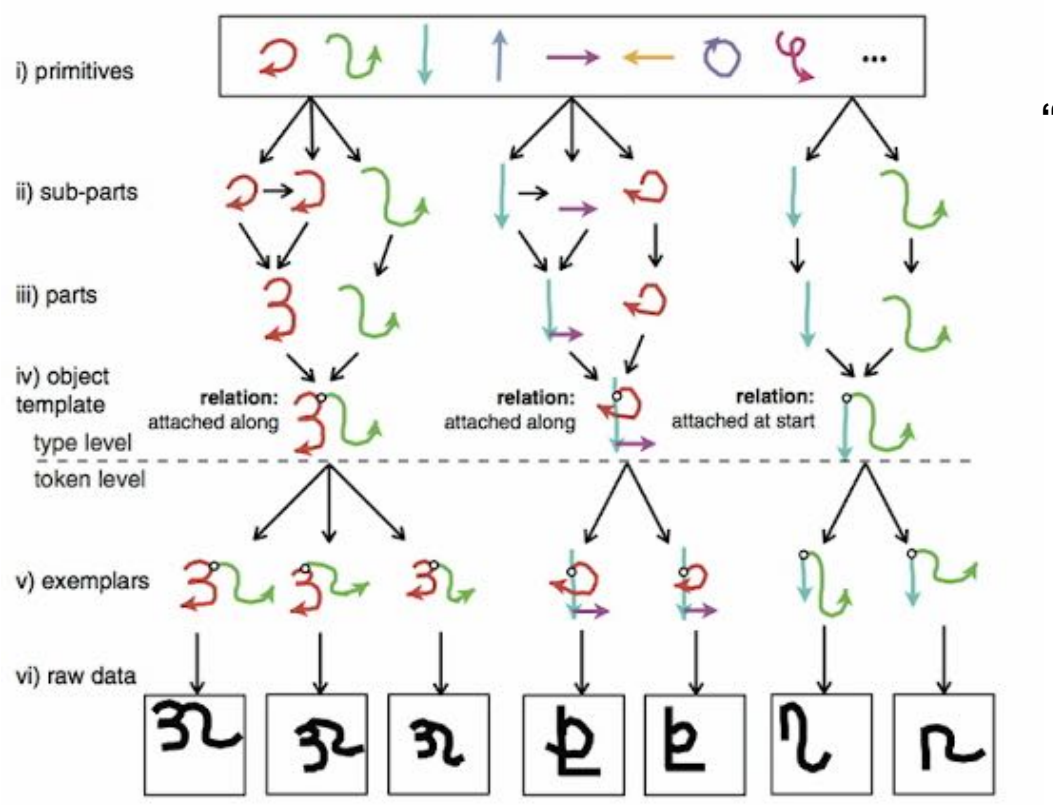
One-shot learning (2/3)

- How would you draw this character?



One-shot learning (3/3)

- Bayesian Program Learning



"Probabilistic Program"

Casual process

Bayesian learning

Commonsense scene understanding

Today's best computer vision systems:
"A child playing with blocks"

"A child sitting on a table"

"A child in a room"

Human-like artificial intelligence

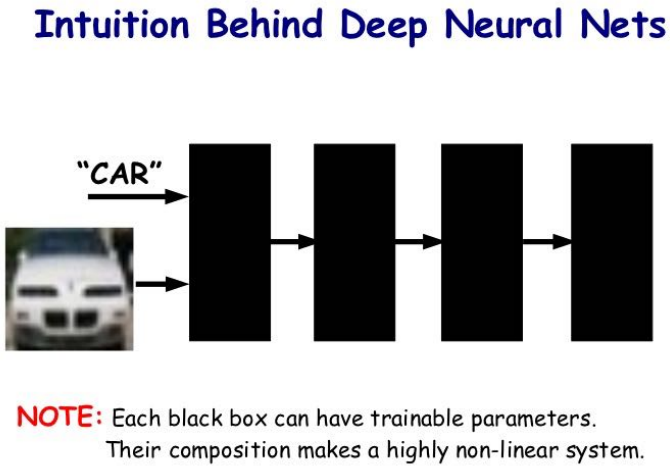
- We can see model-building at work in all the ways that even a young child is more intelligence than any current robot or AI system:
  - Zero-shot & One-shot learning
  - Commonsense scene understanding

- Probabilistic program, Bayesian learning are beginning to let us capture these capacities in machines.
- Human-like artificial intelligence - Build a machine that starts like a baby, and learns like a child. It's the only known scaling path to intelligence that actually works.

LIMITATION OF deep learning (1/2)

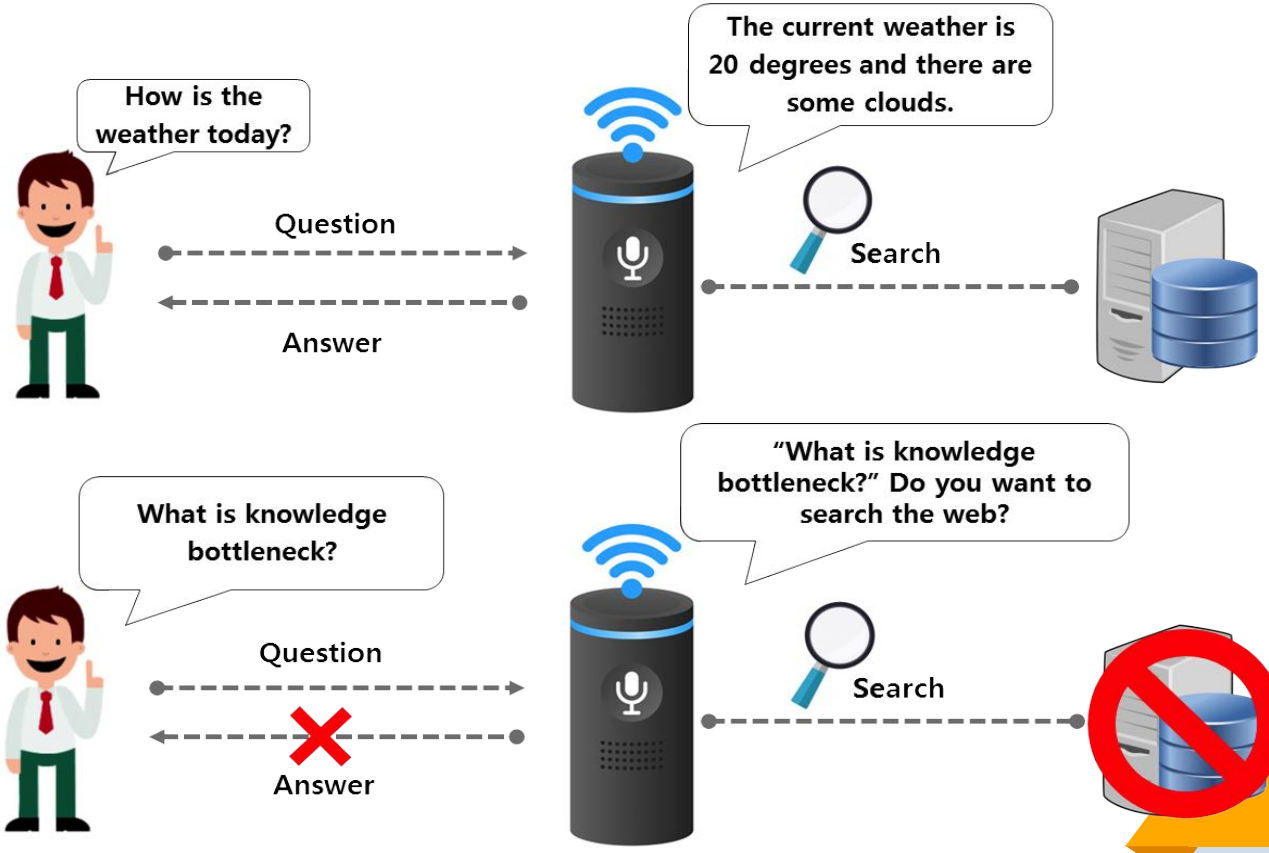
- Lack of explainability and interpretability
- An artificial intelligence model that can not be interpreted and explained.

- Deep learning has a good performance, but there is a disadvantage as a decision making tool (unknown decision making process)
- In order to resolve this, a form that can be understood by a person is required.



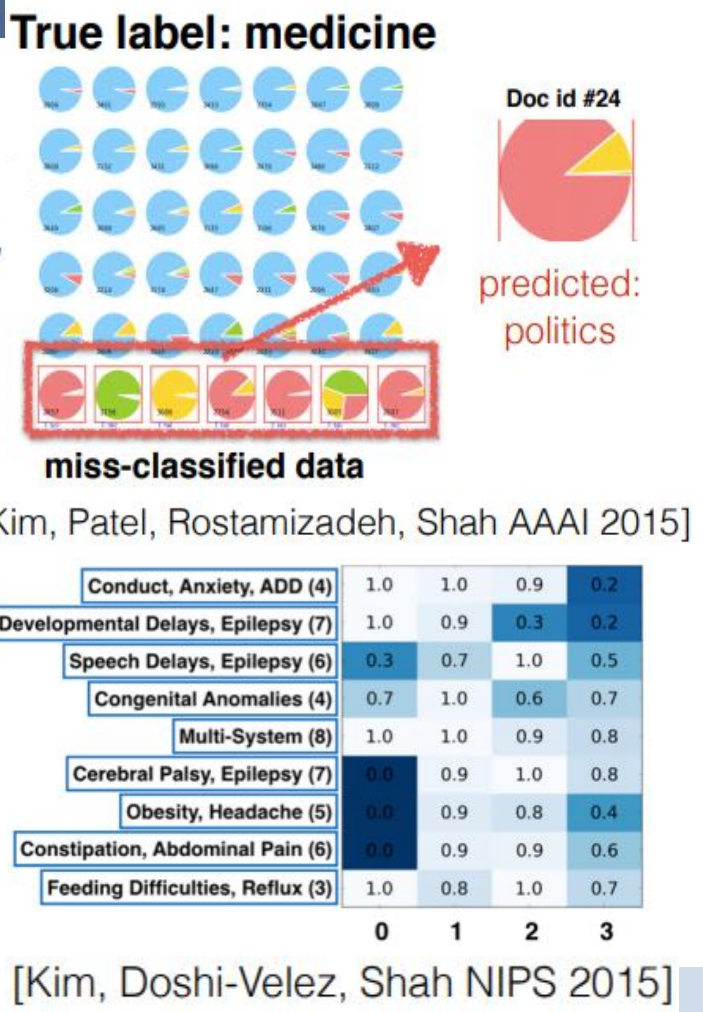
LIMITATION OF deep learning (2/2)

- Knowledge bottleneck
- Hardness of information extraction



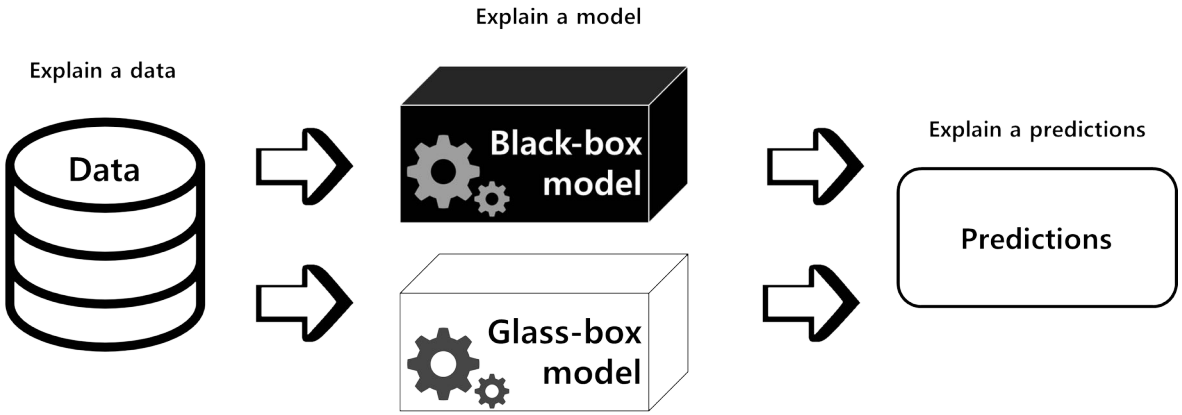
Attempts to resolve (1/)

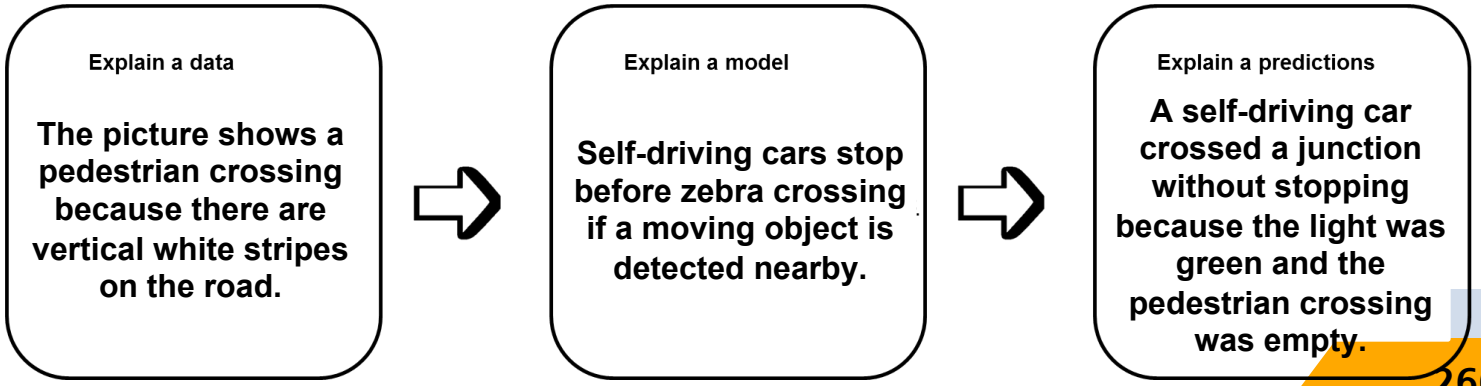
- Interpretability for data exploration
  - Visualization
- Domain specific interpretability
  - Learning features that distinguishes clusters
- Interactive machine learning for debugging models or hyper parameter explorations



Attempts to resolve (2/)

- NLP-based knowledge reasoning and explanatory AI on behavioral results





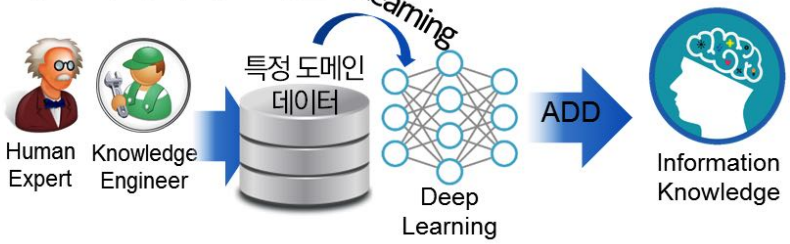
Attempts to resolve (3/)

직접 적용 불가

문제점

- 지식 확장(Information Extraction) 어려움
- AI기술만으로 지식 확장이 느림
- 사람의 기획,승인,검토 필요

기존지식획득AI기술

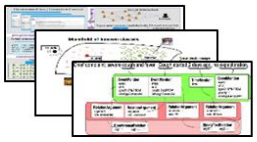


적용

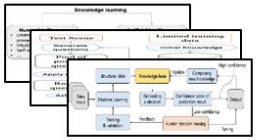
현실 문제

- 지식 이 여러 플랫폼에 흩어져 있음
- 지식 구분 어려움
- 비정형 데이터 분석의 어려움
- 지식이 플랫폼에 의존적

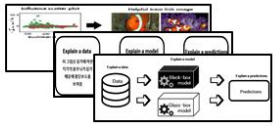
제안기술 : 지식 확장 및 성능 향상 기술 개발



자동 지식 획득 및 추론 엔진



Human Collaboration을 통한 지식 생성, 추론 연구



NLP 기반 지식 추론 및 설명력 있는 AI 기술 개발

직접 적용 및 실증

대화 시스템

- 지식 병목 문제해결과 지식 확장을 위한 인공지능 원천기술 개발

Future Work

Hybrid Approach with Deep Learning Models

Neural Network Based End-to-End Learning Approach

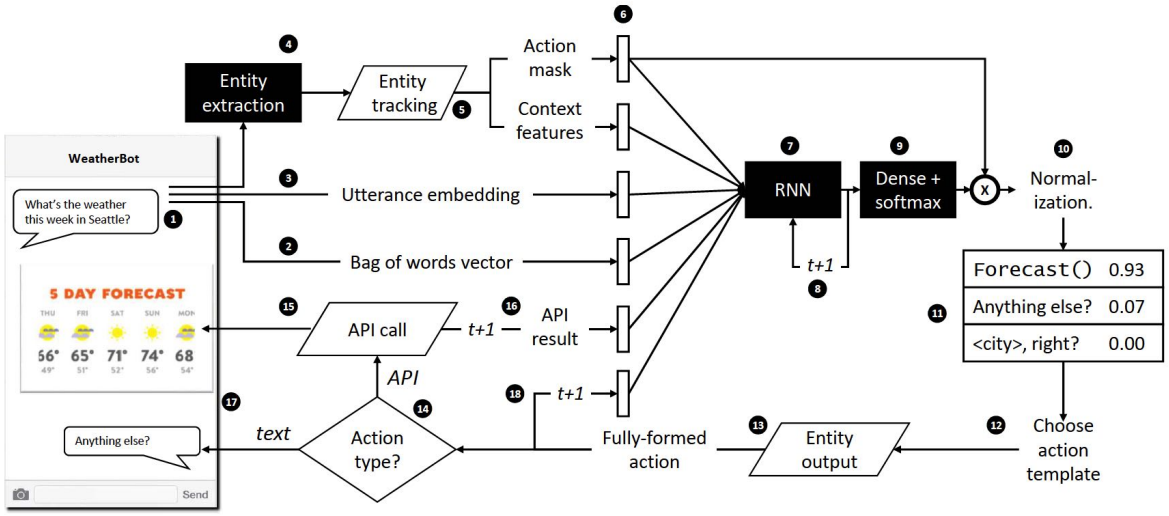
- Learning model can be learned only by learning data itself without knowledge of domain knowledge or hand-craft feature
- Simple data preprocessing
- However, a lot of learning data is required for training.

Use Domain Knowledge with End-to-End Neural Network Approach

- By using knowledge of the domain appropriately, it is possible to learn the model with a relatively small amount of learning data compared to general end-to-end learning models
- Also has end-to-end approach

In Goal-Oriented Dialogue System

- Microsoft proposed a model of a hybrid code network that combines neural network models with domain specific knowledge.
- This system is a conversation system for restaurant reservation purpose.



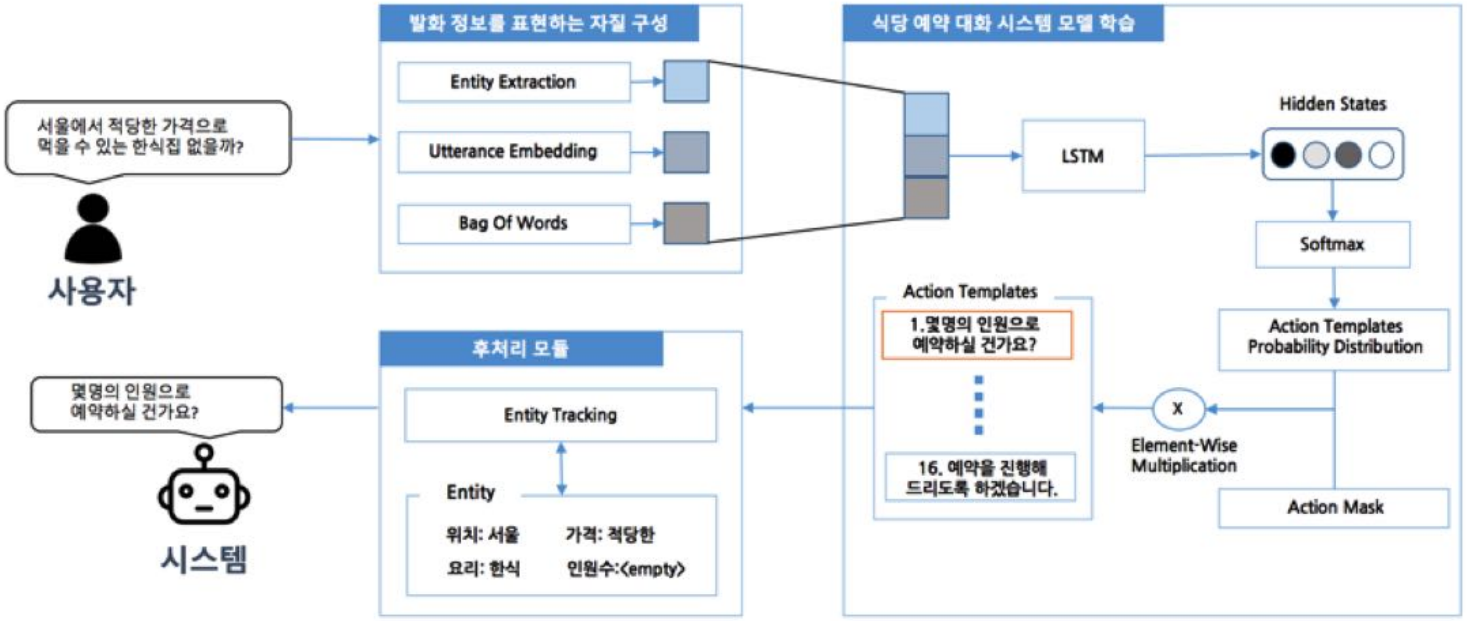
Define Action Templates to Reflect Domain Specific Knowledge in Korean

- Define the response of the conversation system as an action template to represent domain knowledge

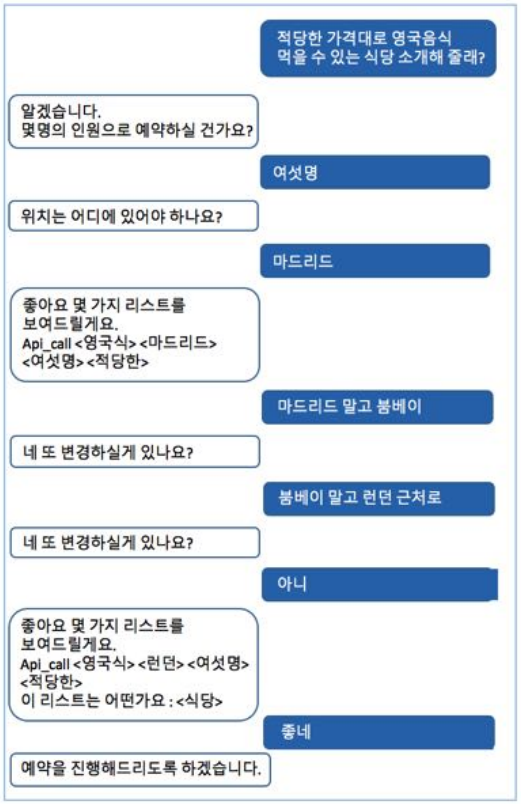
Action Templates

0. api_call <음식종류> <위치> <인원수> <가격>
1. 가격의 범위는 어느정도로 생각하세요
2. 감사합니다
3. 네 또 변경하실게 있나요
4. 다른 리스트를 보여드릴게요
5. 또 도와드릴게 있나요
6. 몇명의 인원으로 예약하실 건가요
7. 안녕하세요 어떻게 도와드릴까요
8. 알겠습니다
9. 어떤 종류의 요리를 좋아하나요
10. 예약을 진행해드리도록 하겠습니다
11. 위치는 <info_address> 입니다
12. 위치는 어디에 있어야 하나요
13. 이 리스트는 어떤가요: <restaurant>
14. 전화번호는 <info_phone> 입니다
15. 좋아요 몇 가지 리스트를 보여드릴게요

Korean Restaurant Reservation System using Hybrid Code Network



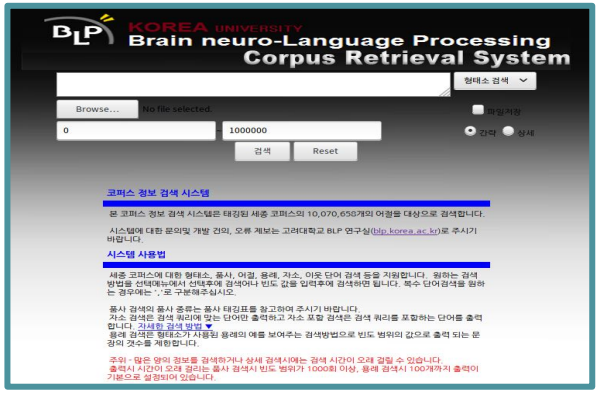
Korean Restaurant Reservation System using Hybrid Code Network



- In the user's first utterance, there is no number of persons and location information required for proceeding the restaurant reservation.
- The system asks the number of people and the location of the restaurant you want to book
- When the necessary attributes are satisfied, th e system calls the api that satisfies the condition
- If the user changes the property value, the syst em calls the api to retrieve the restaurant that sat isfies the conditions of those properties

More Demos

Corpus retrieval system

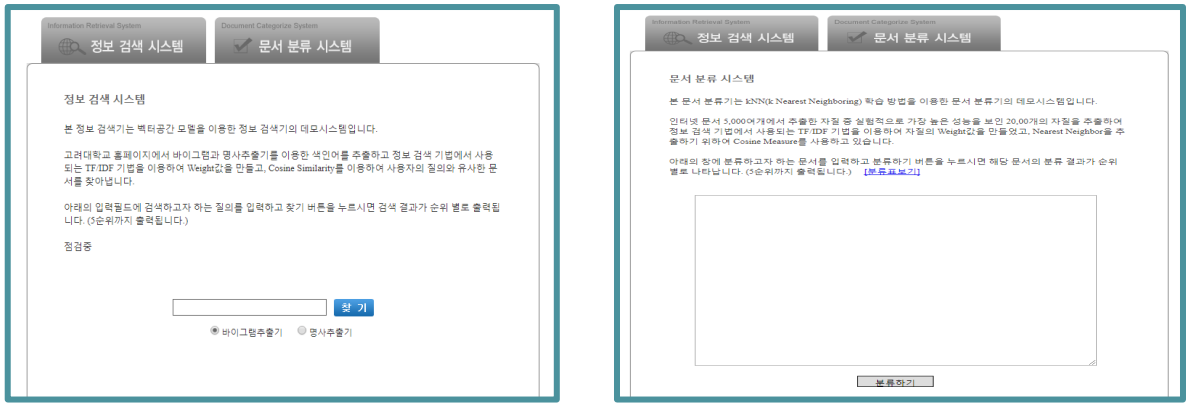


Corpus information retrieval

http://blpdemo.korea.ac.kr/CorpusRetrieval/corpus.php

Supports Sejong corpus for morpheme, part of speech, phrases, poems, and neighboring words.

Document classifier



Information retrieval system / document classification system

http://blpdemo.korea.ac.kr/DocuCate/doccat.htm

Information retrieval system

Information retrieval using vector space model

Extract index words using bigram and noun extractor,

Using tf / idf technique, weights are created, and similar documents are found by using cosine similarity.

Document classification system

Document classifier used KNN (K-Nearest Neighboring)

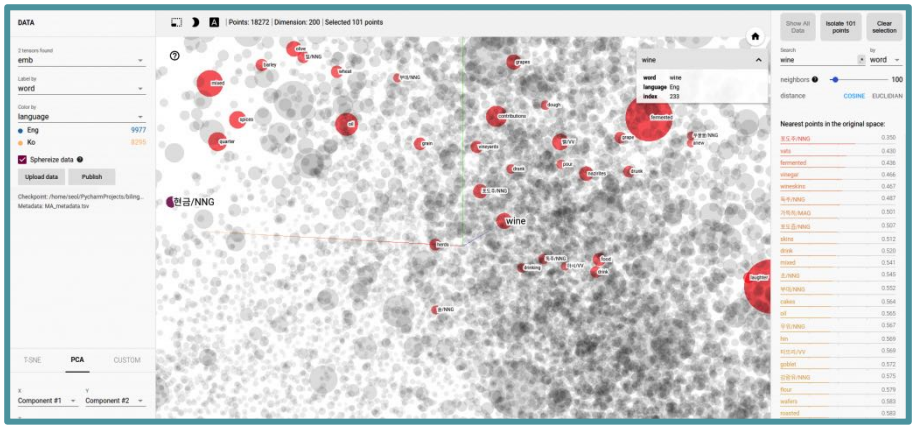
Korean language typing using brain computer interface with predictive models

Brain computer Korean input method using language prediction model

https://youtu.be/02Vv1FAkiQ8

Language prediction model using EEG

Bilingual word embedding



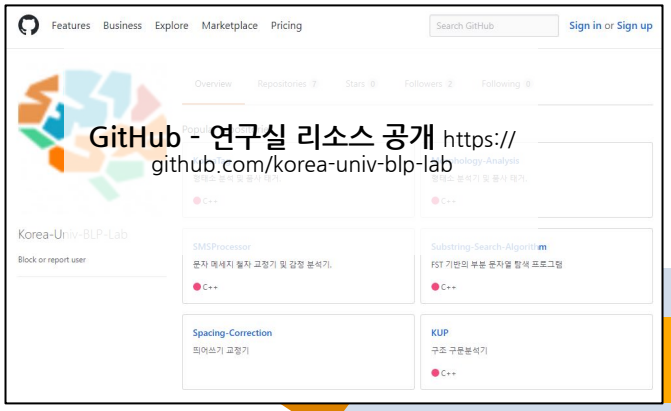
Bilingual Word Embedding

http://blplab.iptime.org:4321/seol2/mt/projector.html
Bilingual word embedding using parallel corpus

These research goal make seed lexicon by aligned parallel data and make mapping function between seed lexicon and general-domain corpora such as Wikipedia.

NLP resources

- Korea university NLP&AI Lab resources
  - Url : github.com/korea-univ-blp-lab
  - Resources of NLP (C++ code)
  - Morphological Analysis and Part-of-speech tagging
  - Text message spelling corrector and emotion analyzer
  - Spacing corrector
  - FST based substring search program
  - Structure syntactic parser



GitHub - 연구실 리소스 공개 https://github.com/korea-univ-blp-lab

THANK YOU

Any questions?