

Artificial Intelligence for Text Analytics

Multilingual Named Entity Recognition (NER), Text Similarity and Clustering

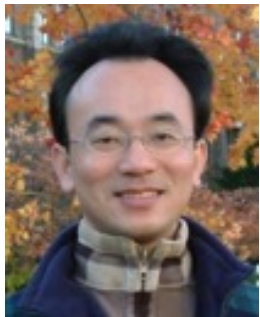
1102AITA06

MBA, IM, NTPU (M5026) (Spring 2022)

Tue 2, 3, 4 (9:10-12:00) (B8F40)



<https://meet.google.com/paj-zhhj-mya>



Min-Yuh Day, Ph.D,
Associate Professor

Institute of Information Management, National Taipei University

<https://web.ntpu.edu.tw/~myday>



Syllabus

Week Date Subject/Topics

- | | | |
|----------|-------------------|---|
| 1 | 2022/02/22 | Introduction to Artificial Intelligence for Text Analytics |
| 2 | 2022/03/01 | Foundations of Text Analytics:
Natural Language Processing (NLP) |
| 3 | 2022/03/08 | Python for Natural Language Processing |
| 4 | 2022/03/15 | Natural Language Processing with Transformers |
| 5 | 2022/03/22 | Case Study on Artificial Intelligence for Text Analytics I |
| 6 | 2022/03/29 | Text Classification and Sentiment Analysis |

Syllabus

Week	Date	Subject/Topics
------	------	----------------

7	2022/04/05	Tomb-Sweeping Day (Holiday, No Classes)
---	------------	---

8	2022/04/12	Midterm Project Report
---	------------	------------------------

9	2022/04/19	Multilingual Named Entity Recognition (NER), Text Similarity and Clustering
---	------------	--

10	2022/04/26	Text Summarization and Topic Models
----	------------	-------------------------------------

11	2022/05/03	Text Generation
----	------------	-----------------

12	2022/05/10	Case Study on Artificial Intelligence for Text Analytics II
----	------------	---

Syllabus

Week Date Subject/Topics

13 2022/05/17 Question Answering and Dialogue Systems

**14 2022/05/24 Deep Learning, Transfer Learning,
Zero-Shot, and Few-Shot Learning for Text Analytics**

15 2022/05/31 Final Project Report I

16 2022/06/07 Final Project Report II

17 2022/06/14 Self-learning

18 2022/06/21 Self-learning

Multilingual Named Entity Recognition (NER) Text Similarity and Clustering

Outline

- **Multilingual Named Entity Recognition (NER)**
- **Text Similarity and Clustering**

Multilingual Named Entity Recognition (NER)

Named Entity Recognition (NER)

- **Named entities**
 - **represent real-world objects**
 - **people, places, organizations**
 - **proper names**
- **Named entity recognition**
 - **Entity chunking**
 - **Entity extraction**

Named Entity Recognition (NER)

Michael Jeffrey Jordan was born in Brooklyn, New York.

$\langle w_1, w_3, \text{Person} \rangle$ Michael Jeffrey Jordan

$\langle w_7, w_7, \text{Location} \rangle$ Brooklyn

$\langle w_9, w_{10}, \text{Location} \rangle$ New York

$\uparrow \langle I_s, I_e, t \rangle$

Named Entity Recognition

$\uparrow s = \langle w_1, w_2, \dots, w_N \rangle$

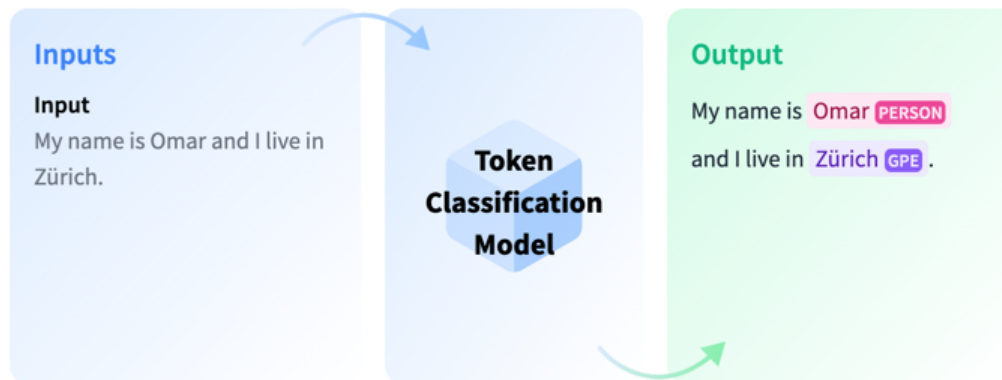
Michael Jeffrey Jordan was born in Brooklyn , New York .
 $w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5 \quad w_6 \quad w_7 \quad w_8 \quad w_9 \quad w_{10} \quad w_{11}$

Token Classification (NER)

[Models](#)[Datasets](#)[Spaces](#)[Docs](#)[Solutions](#)[Pricing](#)[< Tasks](#)

Token Classification

Token classification is a natural language understanding task in which a label is assigned to some tokens in a text. Some popular token classification subtasks are Named Entity Recognition (NER) and Part-of-Speech (PoS) tagging. NER models could be trained to identify specific entities in a text, such as dates, individuals and places; and PoS tagging would identify, for example, which words in a text are verbs, nouns, and punctuation marks.



About Token Classification

Available in **auto TRAIN**

Compatible libraries

[Adapter Transformers](#)[Flair](#)[spaCy](#)[Stanza](#)[Transformers](#)

⚡ Token Classification demo

using [dslim/bert-base-NER](#)

 Token Classification

Example 3 

My name is Clara and I live in Berkeley, California.

Compute

Computation time on cpu: cached

My name is Clara PER and I live in Berkeley LOC , California LOC .

 JSON Output

 Maximize

Models for Token Classification

[Browse Models \(1908\)](#)

 [dslim/bert-base-NER](#)

 Token Classification • Updated Sep 5, 2021 • ↓ 262k • ♥ 42

Named Entity Recognition (NER)

```
from transformers import pipeline
import pandas as pd
classifier = pipeline("ner")
text = "My name is Michael and I live in Berkeley, California."
outputs = classifier(text)
pd.DataFrame(outputs)
```

	entity	score	index	word	start	end
0	I-PER	0.998874	4	Michael	11	18
1	I-LOC	0.997050	9	Berkeley	33	41
2	I-LOC	0.999170	11	California	43	53

spaCy

← → ↻ spacy.io/usage

spaCy 🔥 Out now: spaCy v3.2 [USAGE](#) [MODELS](#) [API](#)

GET STARTED

Installation

- Quickstart
- Instructions
- Troubleshooting
- Changelog

Models & Languages

Facts & Figures

spaCy 101

New in v3.0

New in v3.1

New in v3.2

GUIDES

- Linguistic Features
- Rule-based Matching
- Processing Pipelines
- Embeddings & Transformers NEW
- Training Models NEW
- Layers & Model Architectures NEW
- spaCy Projects NEW
- Saving & Loading
- Visualizers

Operating system macOS / OSX Windows Linux

Platform x86 ARM / M1

Package manager pip conda from source

Hardware CPU GPU

Configuration ☐ virtual env ? ☐ train models ?

Trained pipelines

☐ Catalan ☐ Chinese ☐ Danish ☐ Dutch ☒ English

☐ French ☐ German ☐ Greek ☐ Italian ☐ Japanese

☐ Lithuanian ☐ Macedonian ☒ Multi-language

☐ Norwegian Bokmål ☐ Polish ☐ Portuguese

☐ Romanian ☐ Russian ☐ Spanish

Select pipeline for efficiency ? accuracy ?

```
$ pip install -U pip setuptools wheel
$ pip install -U spacy
$ python -m spacy download en_core_web_trf
$ python -m spacy download xx_sent_ud_sm
```


NER: OntoNotes 5 Named Entities (18)

SID	TYPE	DESCRIPTION
1	PERSON	People, including fictional.
2	NORP	Nationalities or religious or political groups.
3	FAC	Buildings, airports, highways, bridges, etc.
4	ORG	Companies, agencies, institutions, etc.
5	GPE	Countries, cities, states.
6	LOC	Non-GPE locations, mountain ranges, bodies of water.
7	PRODUCT	Objects, vehicles, foods, etc. (Not services.)
8	EVENT	Named hurricanes, battles, wars, sports events, etc.
9	WORK_OF_ART	Titles of books, songs, etc.
10	LAW	Named documents made into laws.
11	LANGUAGE	Any named language.
12	DATE	Absolute or relative dates or periods.
13	TIME	Times smaller than a day.
14	PERCENT	Percentage, including "%".
15	MONEY	Monetary values, including unit.
16	QUANTITY	Measurements, as of weight or distance.
17	ORDINAL	"first", "second", etc.
18	CARDINAL	Numerals that do not fall under another type.

NER: Wikipedia Named Entities

SID	TYPE	DESCRIPTION
1	PER	Named person or family.
2	LOC	Name of politically or geographically defined location (cities, provinces, countries, international regions, bodies of water, mountains).
3	ORG	Named corporate, governmental, or other organizational entity.
4	MISC	Miscellaneous entities, e.g. events, nationalities, products or works of art.

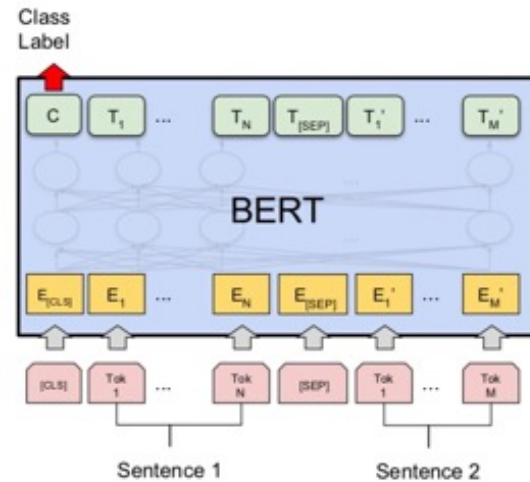
NER IOB Scheme

TAG	ID	DESCRIPTION
"I"	1	Token is inside an entity.
"O"	2	Token is outside an entity.
"B"	3	Token begins an entity.
""	0	No entity tag is set (missing value).

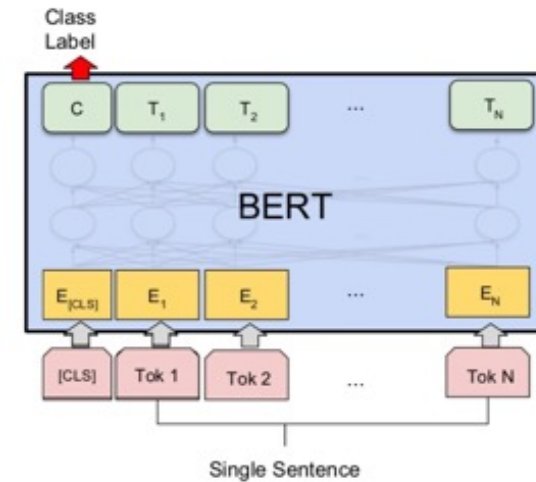
NER **BILUO** Scheme

TAG	DESCRIPTION
BEGIN	The first token of a multi-token entity.
IN	An inner token of a multi-token entity.
LAST	The final token of a multi-token entity.
UNIT	A single-token entity.
OUT	A non-entity token.

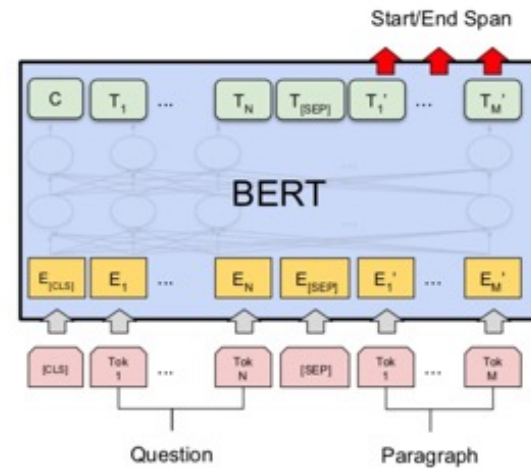
Fine-tuning BERT on NLP Tasks



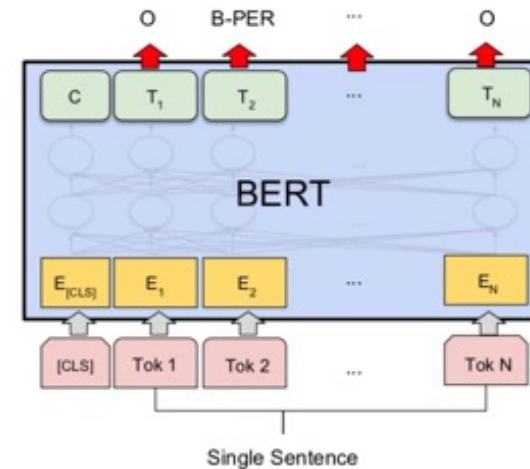
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1

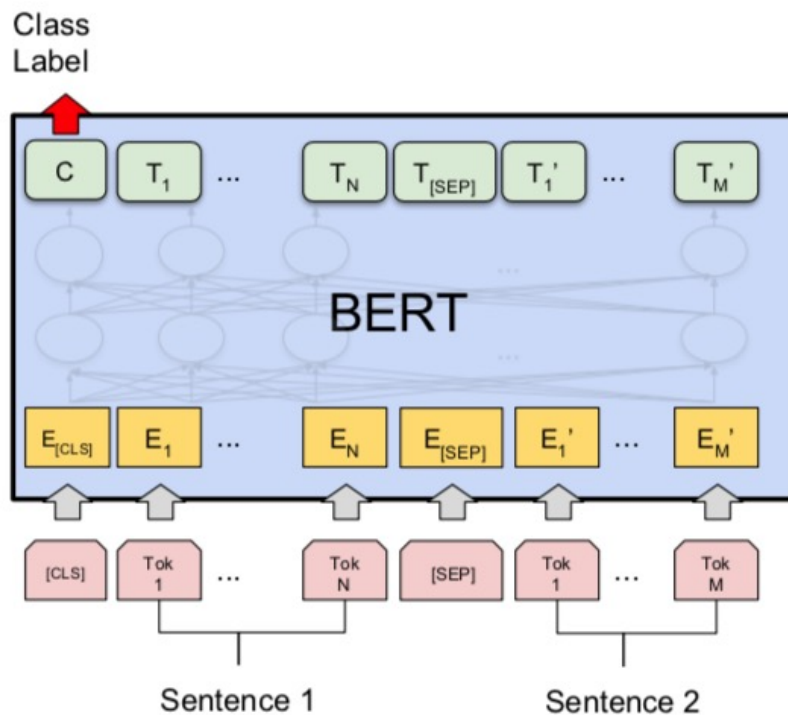


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

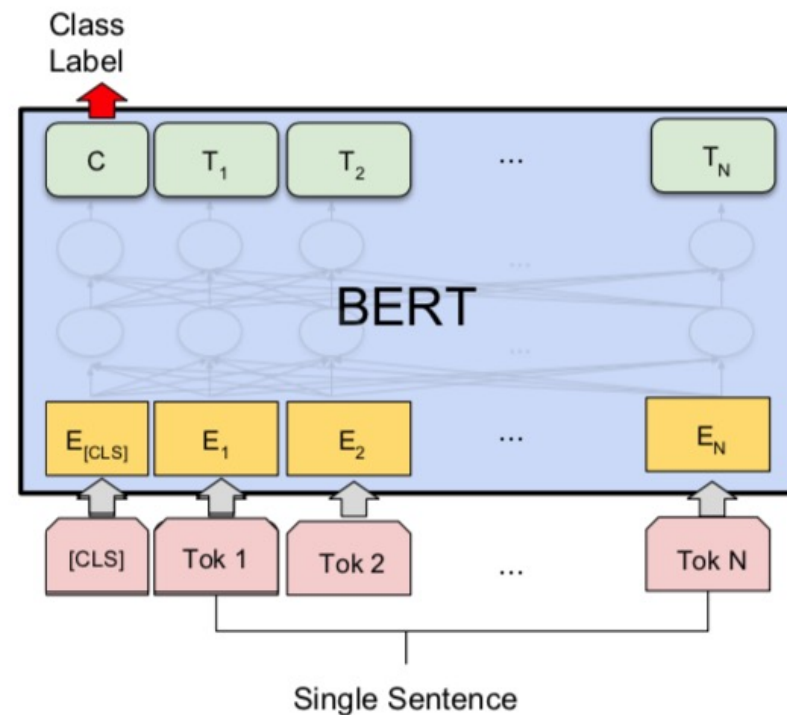
Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." arXiv preprint arXiv:1810.04805

BERT Sequence-level tasks

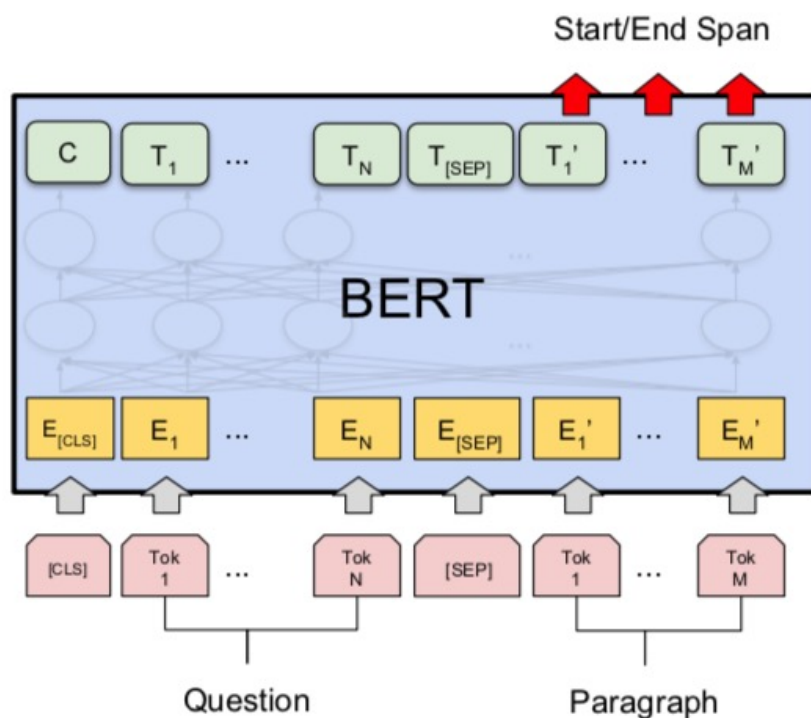


(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG

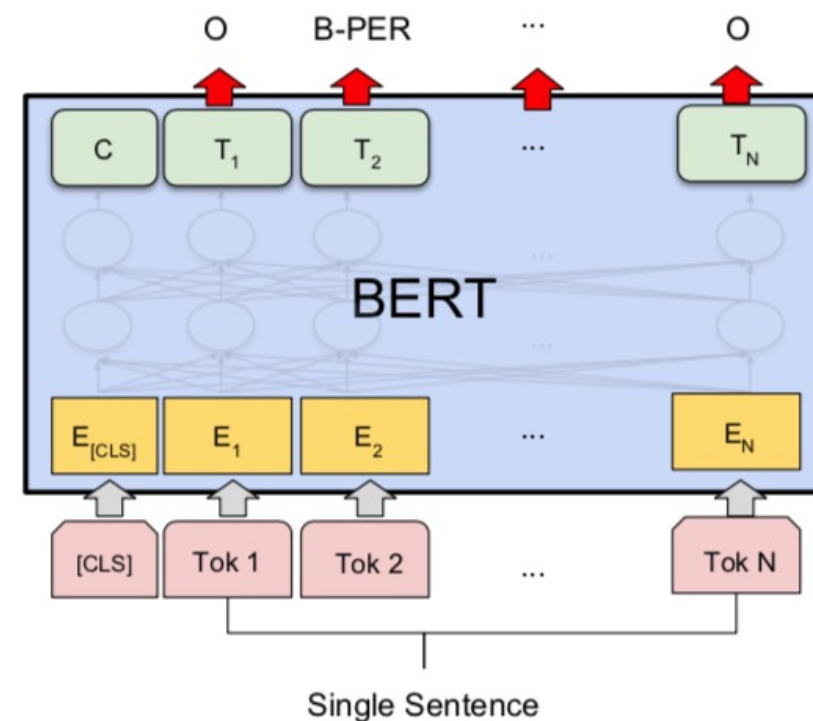


(b) Single Sentence Classification Tasks:
SST-2, CoLA

BERT Token-level tasks



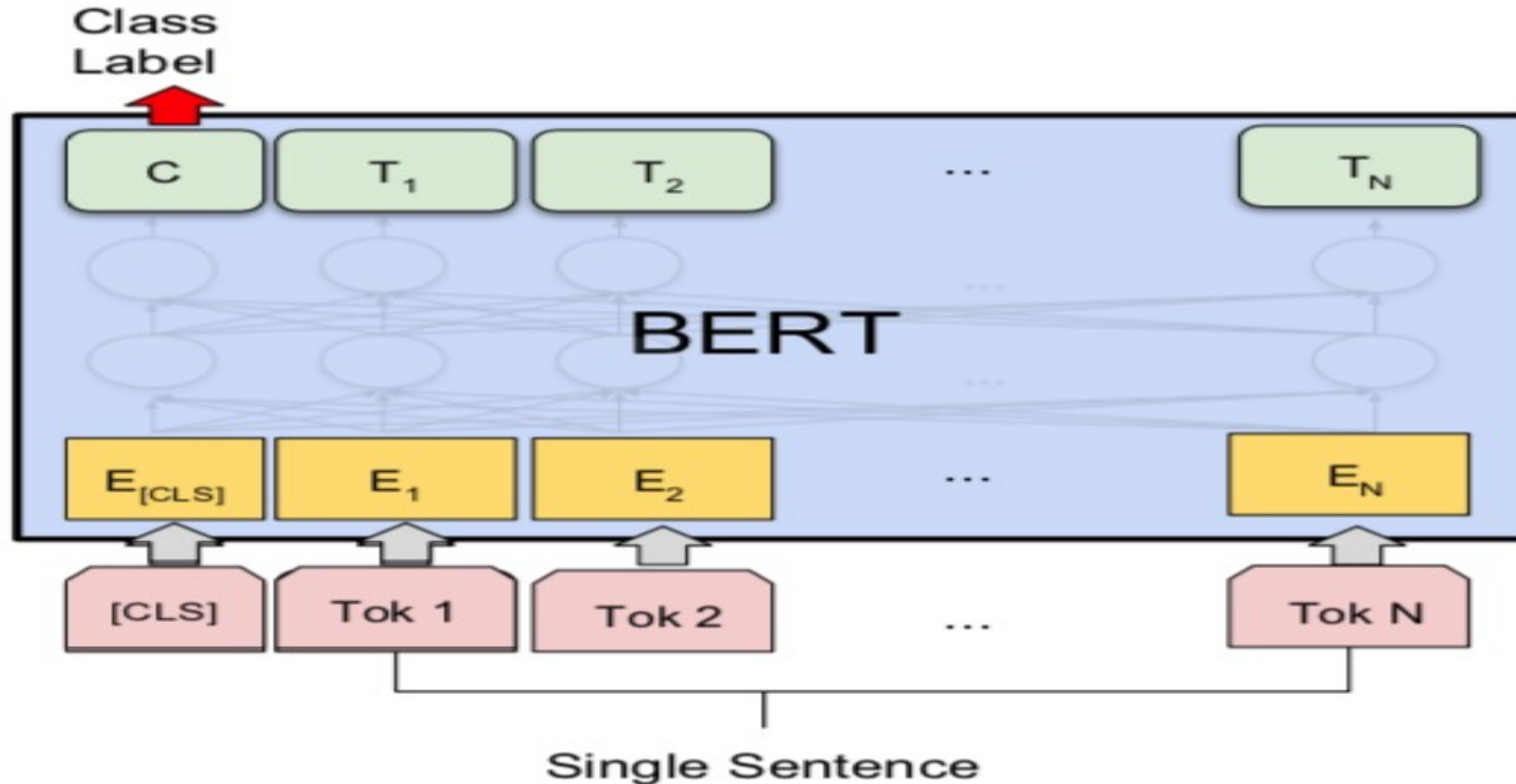
(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

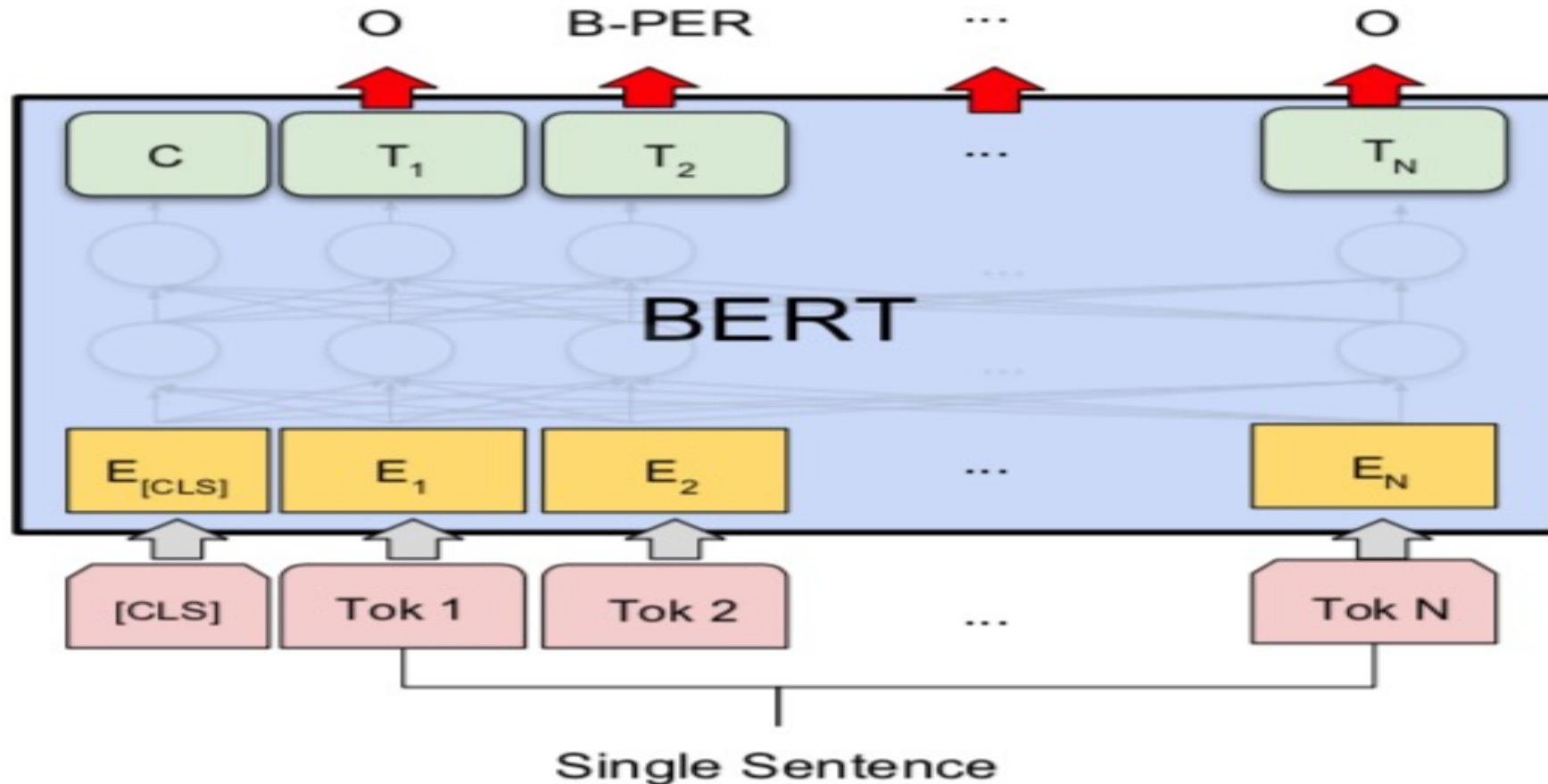
Sentiment Analysis:

Single Sentence Classification



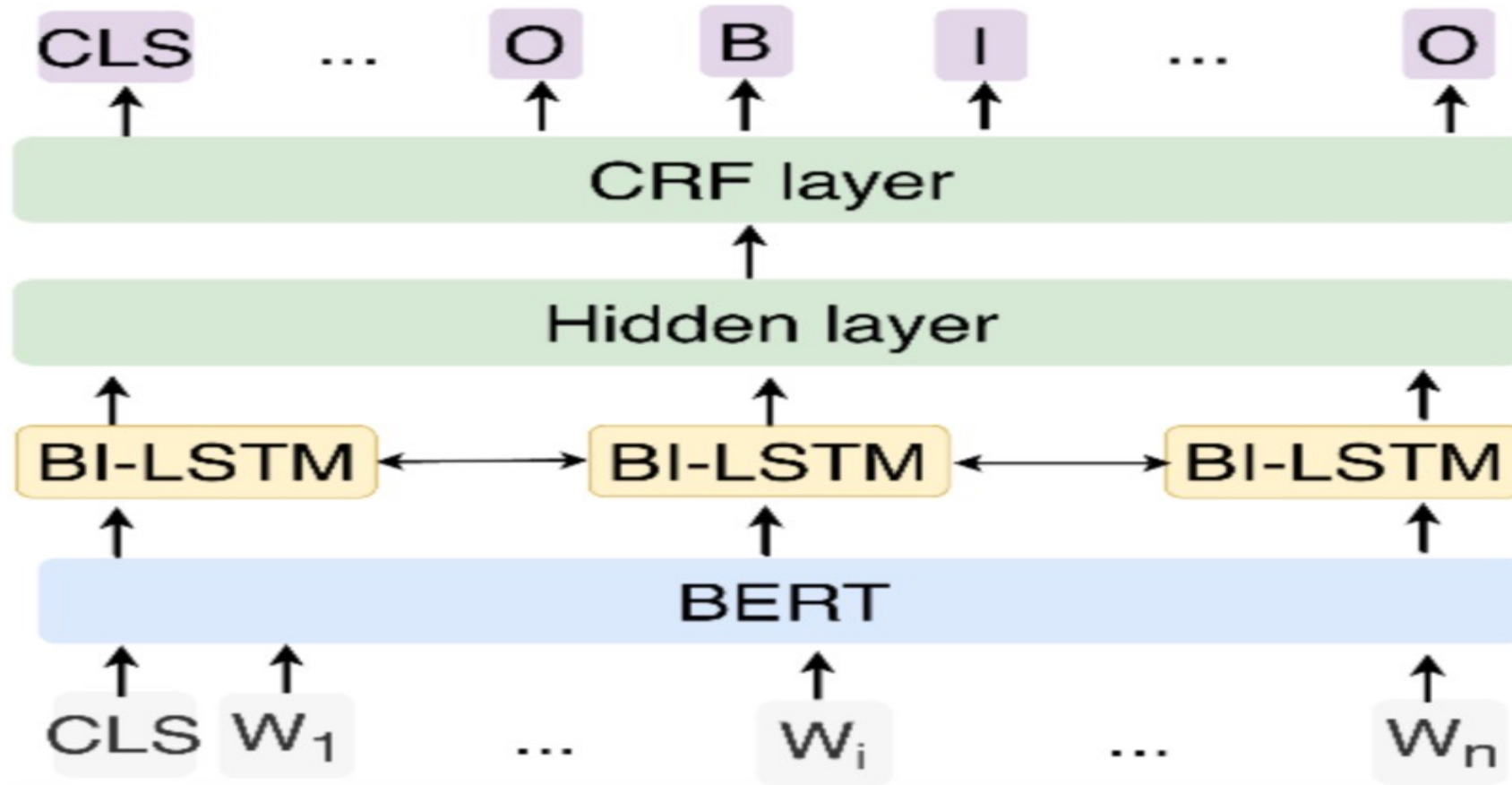
(b) Single Sentence Classification Tasks:
SST-2, CoLA

NER: Single Sentence Tagging



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

NER: Fine-tuning BERT with Bi-LSTM CRF

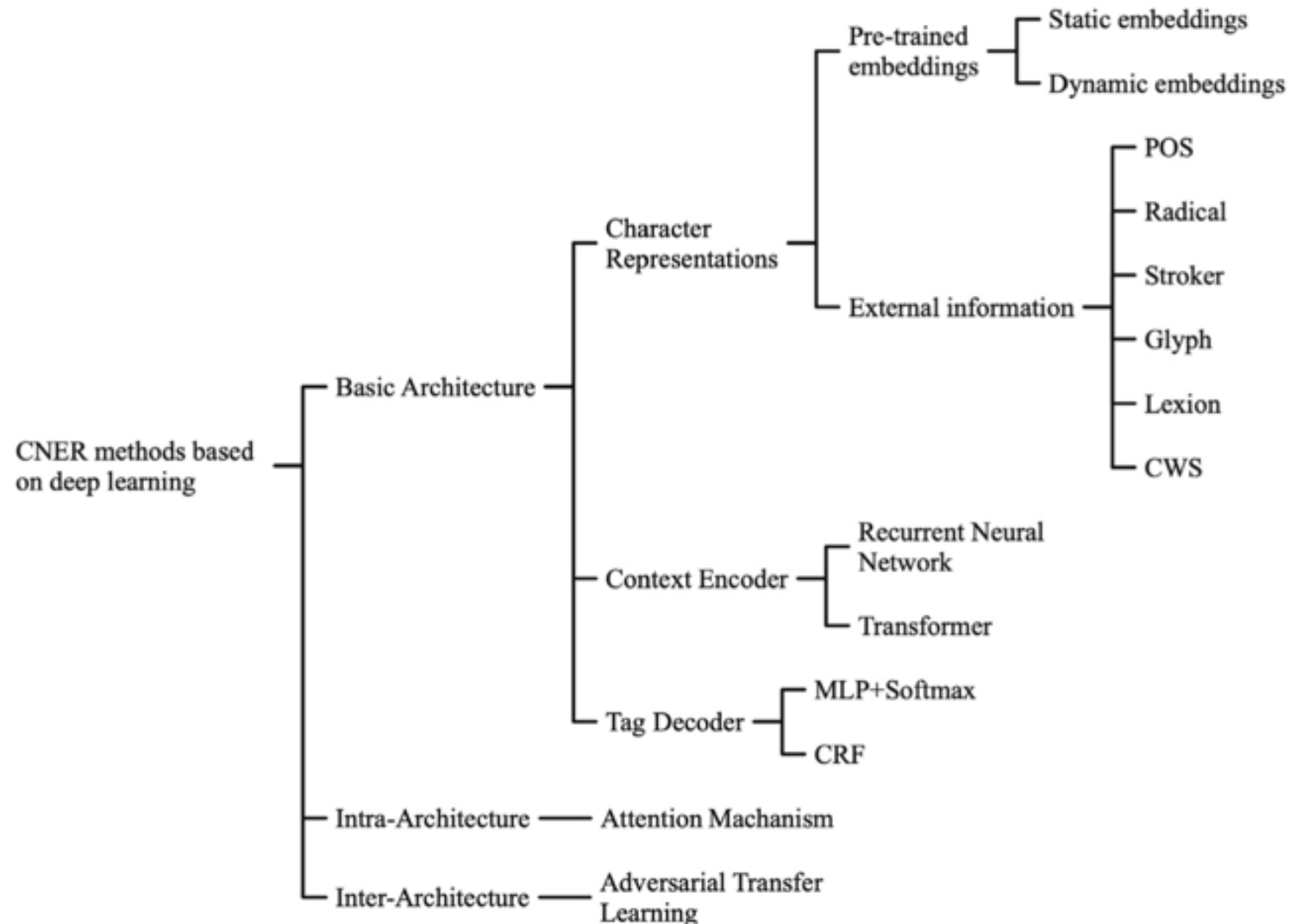


Named Entity Recognition (NER)

Statistical-based methods and **Deep learning-based** methods

	Statistical-based methods	Deep learning-based methods
Character Representations	Handcrafted features (orthographic, prefixes, suffixes, etc.)	Distributed representations (Word2vec, RNN, ELMo, BERT, etc.)
Machine learning models	Statistical-based models (HMM, ME, CRF, SVM, etc.)	Encoder (LSTM, GRU, Transformer, etc.) Decoder (CRF, Transformer, etc.)

The taxonomy of CNER methods based on deep learning



List of Annotated Datasets for English NER

Corpus	Year	Text Source	#Tags	URL
MUC-6	1995	Wall Street Journal	7	https://catalog.ldc.upenn.edu/LDC2003T13 ■
MUC-6 Plus	1995	Additional news to MUC-6	7	https://catalog.ldc.upenn.edu/LDC96T10
MUC-7	1997	New York Times news	7	https://catalog.ldc.upenn.edu/LDC2001T02
CoNLL03	2003	Reuters news	4	https://www.clips.uantwerpen.be/conll2003/ner/
ACE	2000 - 2008	Transcripts, news	7	https://www.ldc.upenn.edu/collaborations/past-projects/ace
OntoNotes	2007 - 2012	Magazine, news, web, etc.	18	https://catalog.ldc.upenn.edu/LDC2013T19
W-NUT	2015 - 2018	User-generated text	6/10	http://noisy-text.github.io
BBN	2005	Wall Street Journal	64	https://catalog.ldc.upenn.edu/LDC2005T33
WikiGold	2009	Wikipedia	4	https://figshare.com/articles/Learning_multilingual_named_entity_recognition_from_Wikipedia/5462500
WiNER	2012	Wikipedia	4	http://rali.iro.umontreal.ca/rali/en/winer-wikipedia-for-ner
WikiFiger	2012	Wikipedia	112	https://github.com/xiaoling/figer
HYENA	2012	Wikipedia	505	https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/hyena/
N ³	2014	News	3	http://aksw.org/Projects/N3NERNEDNIF.html
Gillick	2016	Magazine, news, web, etc.	89	https://arxiv.org/e-print/1412.1820v2
FG-NER	2018	Various	200	https://fgner.alt.ai/
NNE	2019	Newswire	114	https://github.com/nickyringland/nested_named_entities
GENIA	2004	Biology and clinical text	36	http://www.geniaproject.org/home
GENETAG	2005	MEDLINE	2	https://sourceforge.net/projects/bioc/files/
FSU-PRGE	2010	PubMed and MEDLINE	5	https://julielab.de/Resources/FSU_PRGE.html
NCBI-Disease	2014	PubMed	1	https://www.ncbi.nlm.nih.gov/CBBresearch/Dogan/DISEASE/
BC5CDR	2015	PubMed	3	http://bioc.sourceforge.net/
DFKI	2018	Business news and social media	7	https://dfki-lt-re-group.bitbucket.io/product-corpus/

“#Tags” refers to the number of entity types.

Source: Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li (2022). "A survey on deep learning for named entity recognition." IEEE Transactions on Knowledge and Data Engineering 34, no. 1 (2022): 50-70.

Named Entity Recognition (NER)

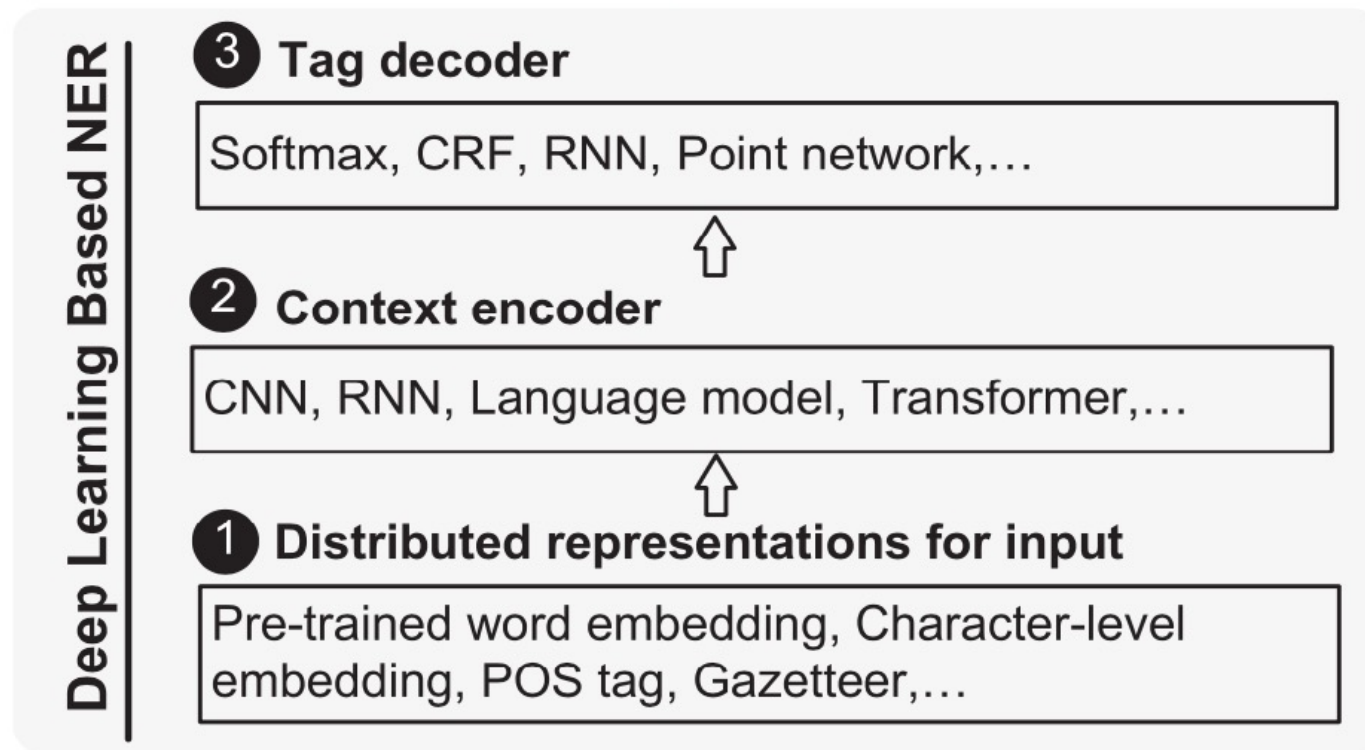
NER System	URL
StanfordCoreNLP	https://stanfordnlp.github.io/CoreNLP/
OSU Twitter NLP	https://github.com/aritter/twitter_nlp
Illinois NLP	http://cogcomp.org/page/software/
NeuroNER	http://neuroner.com/
NERsuite	http://nersuite.nlplab.org/
Polyglot	https://polyglot.readthedocs.io
Gimli	http://bioinformatics.ua.pt/gimli

Named Entity Recognition (NER)

NER System	URL
spaCy	https://spacy.io/api/entityrecognizer
NLTK	https://www.nltk.org
OpenNLP	https://opennlp.apache.org/
LingPipe	http://alias-i.com/lingpipe-3.9.3/
AllenNLP	https://demo.allennlp.org/
IBM Watson	https://natural-language-understanding-demo.ng.bluemix.net/
FG-NER	https://fgner.alt.ai/extractor/
Intellexer	http://demo.intellexer.com/
Repustate	https://repustate.com/named-entity-recognition-api-demo/
AYLIEN	https://developer.aylien.com/text-api-demo
Dandelion API	https://dandelion.eu/semantic-text/entity-extraction-demo/
displaCy	https://explosion.ai/demos/displacy-ent
ParallelDots	https://www.paralldots.com/named-entity-recognition
TextRazor	https://www.textrazor.com/named_entity_recognition

Named Entity Recognition (NER)

B-PER I-PER E-PER O O O S-LOC O B-LOC E-LOC O
Michael Jeffrey Jordan was born in Brooklyn , New York .



Michael Jeffrey Jordan was born in Brooklyn, New York.

Deep Learning for Named Entity Recognition (NER)

- **Distributed Representations for Input**
 - Hybrid Representation
- **Context Encoder Architectures**
 - Deep Transformer
- **Tag Decoder Architectures**
 - Conditional Random Fields (CRF)

Deep Learning for Named Entity Recognition (NER)

- **Distributed Representations for Input**
 - **Word-Level Representation**
 - **Character-Level Representation**
 - **Hybrid Representation**

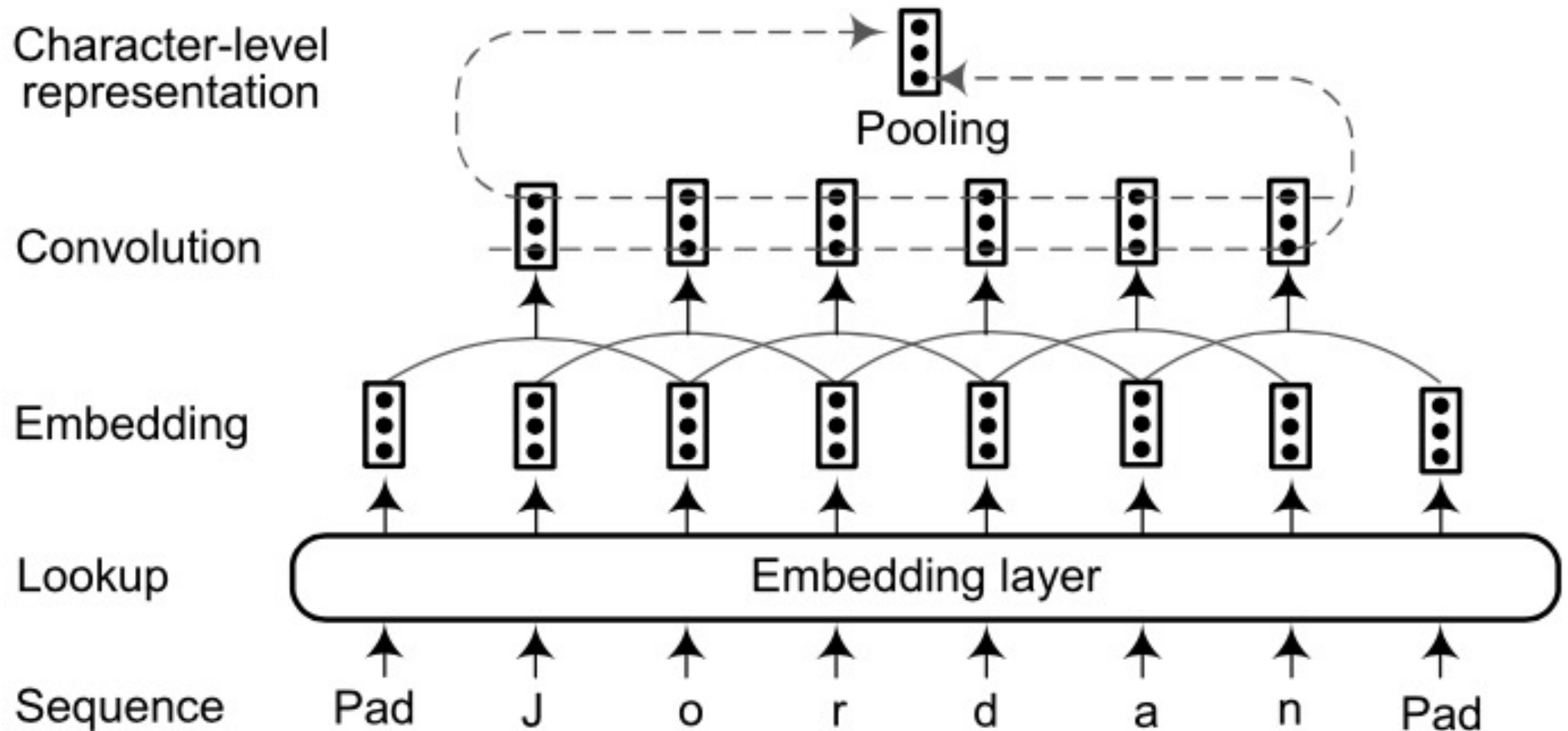
Deep Learning for Named Entity Recognition (NER)

- **Context Encoder Architectures**
 - **Convolutional Neural Networks**
 - **Recurrent Neural Networks**
 - **Recursive Neural Networks**
 - **Neural Language Models**
 - **Deep Transformer**

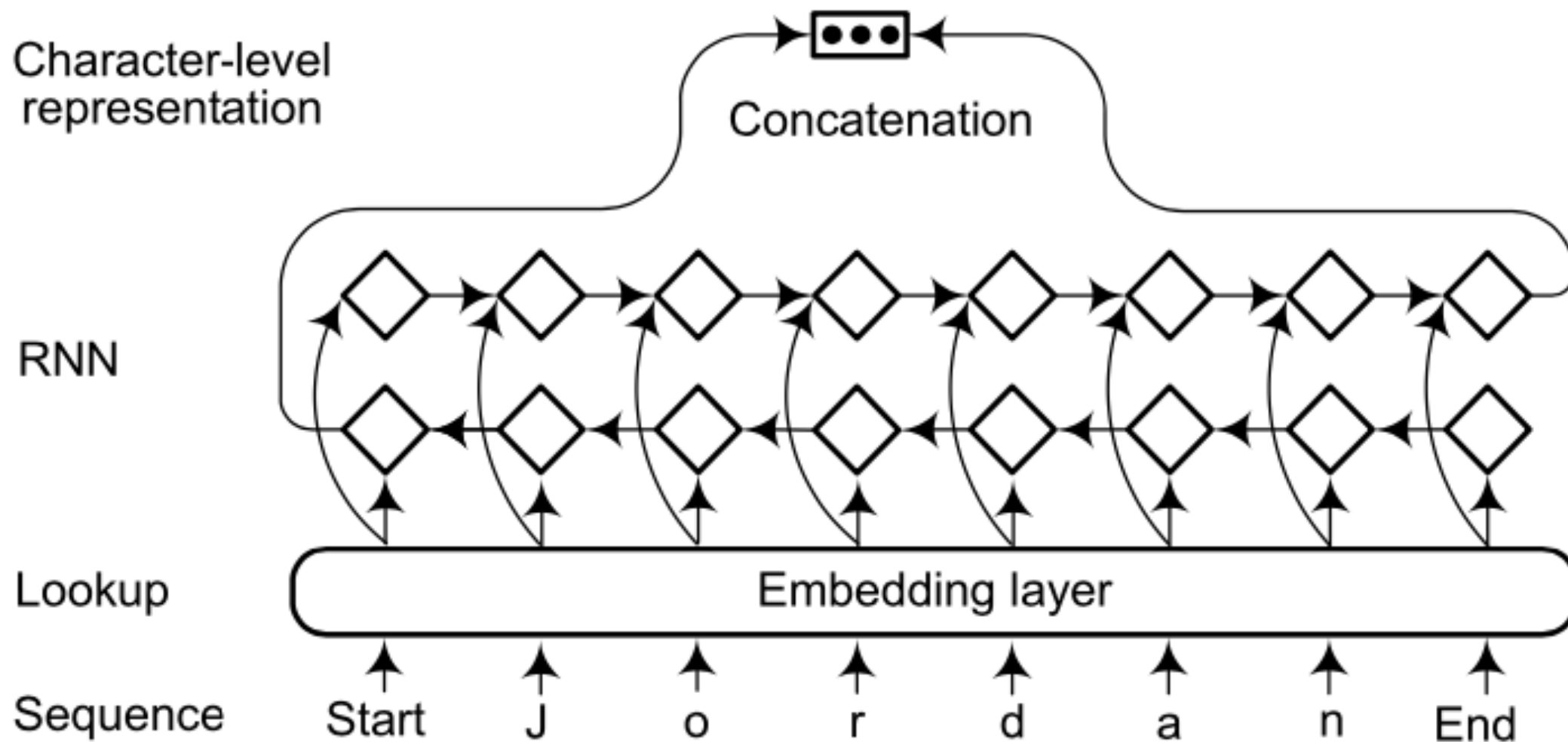
Deep Learning for Named Entity Recognition (NER)

- **Tag Decoder Architectures**
 - **Multi-Layer Perceptron + Softmax**
 - **Conditional Random Fields (CRF)**
 - **Recurrent Neural Networks**
 - **Pointer Networks**

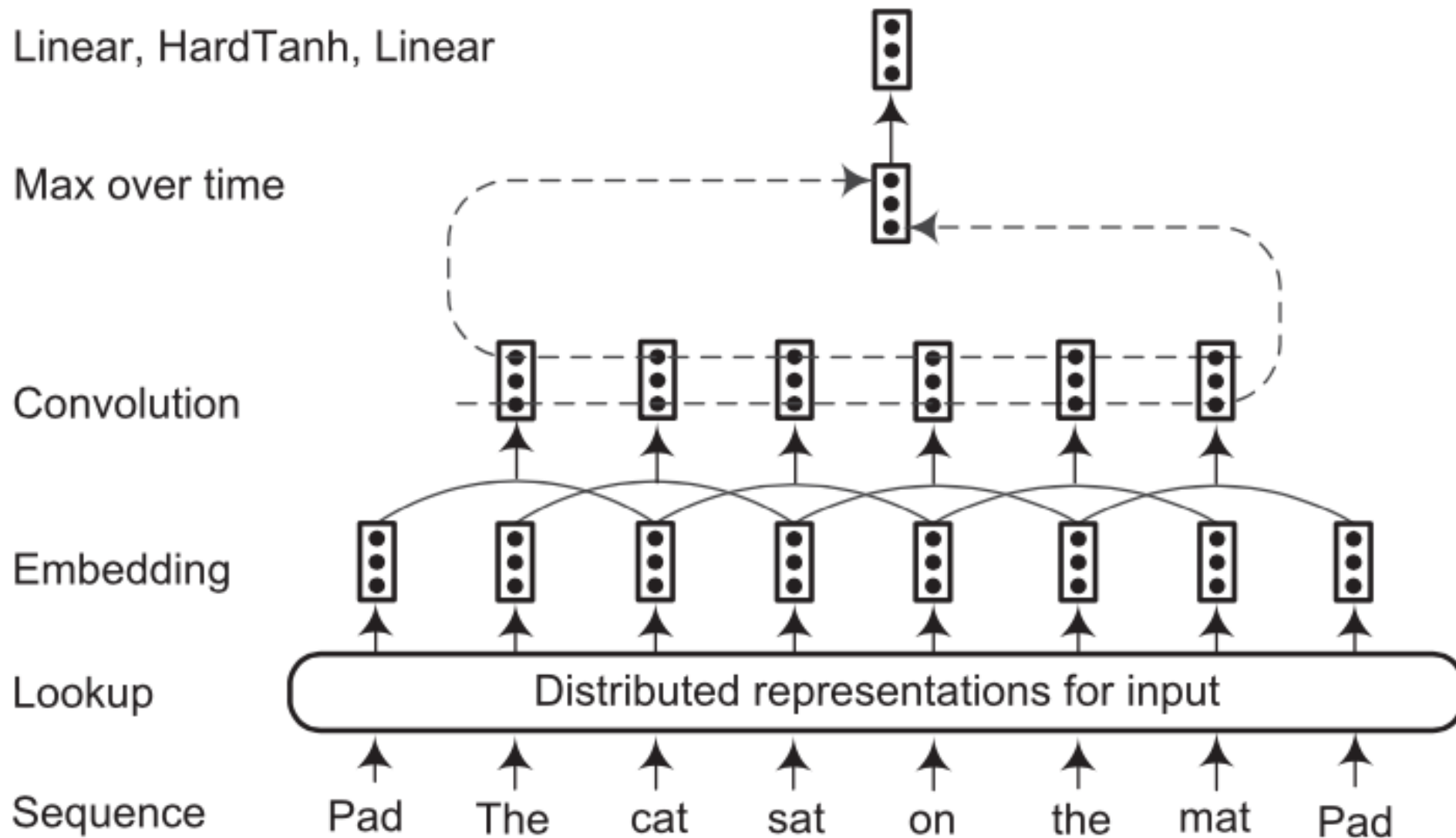
CNN-based character-level representation



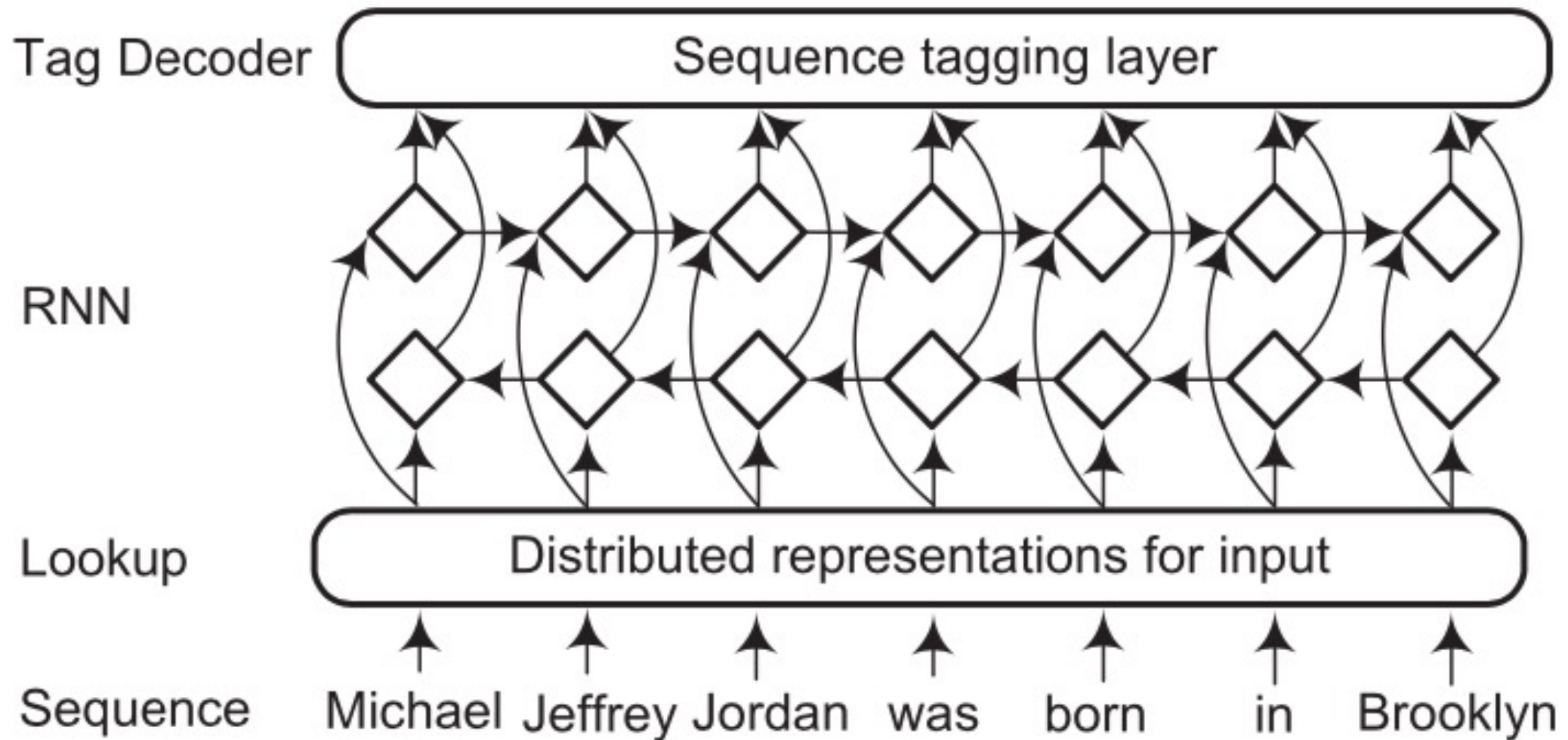
RNN-based character-level representation



Sentence approach network based on CNN

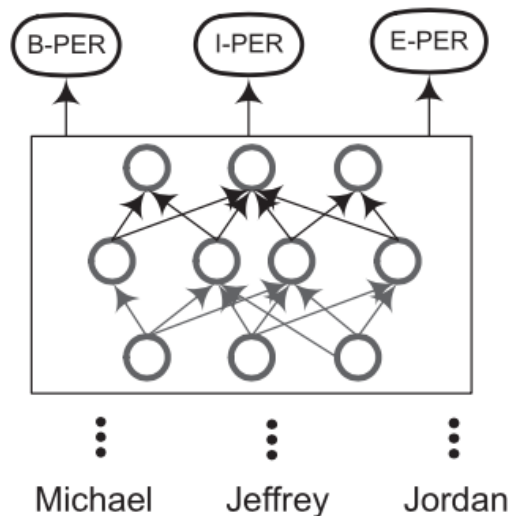


The architecture of RNN-based context encoder

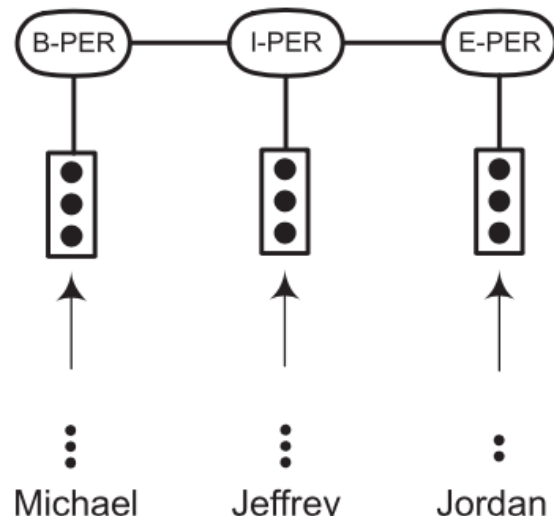


Named Entity Recognition (NER)

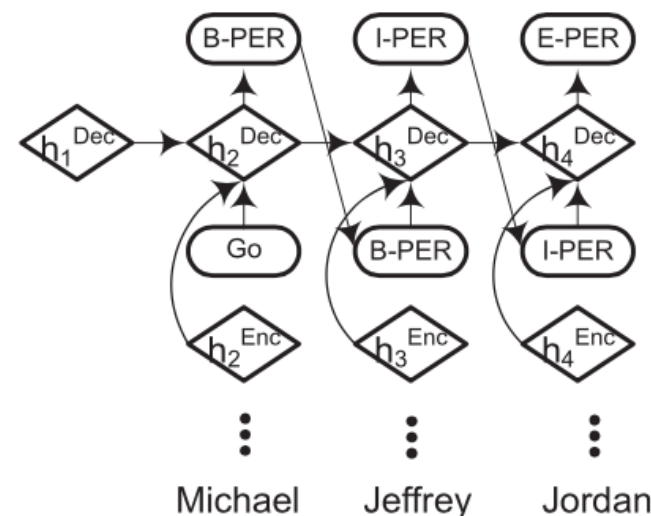
Four tag decoders: MLP+Softmax, CRF, RNN, and Pointer network



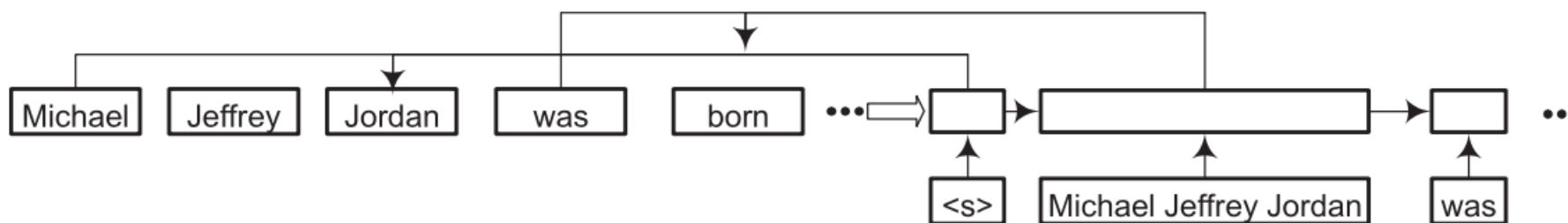
(a) MLP+Softmax



(b) CRF



(c) RNN



(d) Pointer Network

Named Entity Recognition (NER)

Work	Input representation			Context encoder	Tag decoder	Performance (F-score)
	Character	Word	Hybrid			
[93]	-	Trained on PubMed	POS	CNN	CRF	GENIA: 71.01%
[88]	-	Trained on Gigaword	-	GRU	GRU	ACE 2005: 80.00%
[94]	-	Random	-	LSTM	Pointer Network	ATIS: 96.86%
[89]	-	Trained on NYT	-	LSTM	LSTM	NYT: 49.50%
[90]	-	SENNA	Word shape	ID-CNN	CRF	CoNLL03: 90.65%; OntoNotes5.0: 86.84%
[95]	-	Google word2vec	-	LSTM	LSTM	CoNLL04: 75.0%
[99]	LSTM	-	-	LSTM	CRF	CoNLL03: 84.52%
[96]	CNN	GloVe	-	LSTM	CRF	CoNLL03: 91.21%
[104]	LSTM	Google word2vec	-	LSTM	CRF	CoNLL03: 84.09%
[19]	LSTM	SENNA	-	LSTM	CRF	CoNLL03: 90.94%
[105]	GRU	SENNA	-	GRU	CRF	CoNLL03: 90.94%
[97]	CNN	GloVe	POS	BRNN	Softmax	OntoNotes5.0: 87.21%
[106]	LSTM-LM	-	-	LSTM	CRF	CoNLL03: 93.09%; OntoNotes5.0: 89.71%
[102]	CNN-LSTM-LM	-	-	LSTM	CRF	CoNLL03: 92.22%
[17]	-	Random	POS	CNN	CRF	CoNLL03: 89.86%
[18]	-	SENNA	Spelling, n-gram, gazetteer	LSTM	CRF	CoNLL03: 90.10%
[20]	CNN	SENNA	capitalization, lexicons	LSTM	CRF	CoNLL03: 91.62%; OntoNotes5.0: 86.34%
[115]	-	-	FOFE	MLP	CRF	CoNLL03: 91.17%
[100]	LSTM	GloVe	-	LSTM	CRF	CoNLL03: 91.07%
[112]	LSTM	GloVe	Syntactic	LSTM	CRF	W-NUT17: 40.42%
[101]	CNN	SENNA	-	LSTM	Reranker	CoNLL03: 91.62%
[113]	CNN	Twitter Word2vec	POS	LSTM	CRF	W-NUT17: 41.86%
[114]	LSTM	GloVe	POS, topics	LSTM	CRF	W-NUT17: 41.81%
[117]	LSTM	GloVe	Images	LSTM	CRF	SnapCaptions: 52.4%
[108]	LSTM	SSKIP	Lexical	LSTM	CRF	CoNLL03: 91.73%; OntoNotes5.0: 87.95%

Named Entity Recognition (NER)

Work	Input representation			Context encoder	Tag decoder	Performance (F-score)
	Character	Word	Hybrid			
[118]	-	WordPiece	Segment, position	Transformer	Softmax	CoNLL03: 92.8%
[120]	LSTM	SENNA	-	LSTM	Softmax	CoNLL03: 91.48%
[123]	LSTM	Google Word2vec	-	LSTM	CRF	CoNLL03: 86.26%
[21]	GRU		LM	GRU	CRF	CoNLL03: 91.93%
[125]	LSTM	GloVe	-	LSTM	CRF	CoNLL03: 91.71%
[141]	-	SENNA	POS, gazetteers	CNN	Semi-CRF	CoNLL03: 90.87%
[142]	LSTM	GloVe	-	LSTM	Semi-CRF	CoNLL03: 91.38%
[87]	CNN	Trained on Gigaword	-	LSTM	LSTM	CoNLL03: 90.69%; OntoNotes5.0: 86.15%
[109]	-		ELMo, dependency	LSTM	CRF	CoNLL03: 92.4%; OntoNotes5.0: 89.88%
[107]	CNN	GloVe	ELMo, gazetteers	LSTM	Semi-CRF	CoNLL03: 92.75%; OntoNotes5.0: 89.94%
[132]	LSTM	GloVe	ELMo, POS	LSTM	Softmax	CoNLL03: 92.28%
[136]	-	-	BERT	-	Softmax	CoNLL03: 93.04%; OntoNotes5.0: 91.11%
[137]	-	-	BERT	-	Softmax +Dice Loss	CoNLL03: 93.33%; OntoNotes5.0: 92.07%
[133]	LSTM	GloVe	BERT, document-level embeddings	LSTM	CRF	CoNLL03: 93.37%; OntoNotes5.0: 90.3%
[134]	CNN	GloVe	BERT, global embeddings	GRU	GRU	CoNLL03: 93.47%
[131]	CNN	-	Cloze-style LM embeddings	LSTM	CRF	CoNLL03: 93.5%
[135]	-	GloVe	Pooled contextual embeddings	RNN	CRF	CoNLL03: 93.47%

Applied Deep Learning for Named Entity Recognition (NER)

- **Deep Multi-Task Learning for NER**
- **Deep Transfer Learning for NER**
- **Deep Active Learning for NER**
- **Deep Reinforcement Learning for NER**
- **Deep Adversarial Learning for NER**
- **Neural Attention for NER**

Named Entity Recognition (NER)

Message Understanding Conference (MUC) Corpus

Year	Conf.	Language	Source Type	Data Sources	Task
1987	MUC1	English	Military reports	Fleet Operations	Open ended (no pre-defined template)
1989	MUC2	English	Military reports	Fleet Operations	IE in form of pre-provided template
1991	MUC3	English	Reports from News	Acts of terrorism in Latin America	IE in form of pre-provided template
1992	MUC4	English	Reports from News	Acts of terrorism in Latin America	IE in form of pre-provided template
1993	MUC5	English, Japanese	Reports from News	Corporate Joint Ventures, Microelectronic production	IE in form of pre-provided template
1995	MUC6	English	Reports from News	Negotiation of Labor Disputes and Corporate Management Succession	NER, Coreference Resolution, Description of NEs and scenarios
1997	MUC7	English	Reports from News	Reports on various aerial crashes, launch report of various missiles and rockets	NER, Coreference Resolution, Description of NEs and scenarios

Named Entity Recognition (NER)

Automatic Content Extraction (ACE) corpus

Corpus	Tasks	Language	Data Source
ACE 2002	EDT, RDC	English	Newsire
ACE 2003	EDT, RDC	English	Newsire, Broadcast
	EDT	Arabic	
ACE 2004	EDT, RDC, LNK	English, Arabic, Chinese	Newsire, Broadcast
ACE 2005	EDT, EDC, RDC, LNK, Time-Stamping	English, Chinese	Newsire, Newsgroups, Weblogs Broadcast
	EDT, EDC, RDC, LNK	Arabic	
ACE 2007	EDT, EDC, RDC, LNK	Arabic, Spanish	Newsire, Weblogs

Named Entity Recognition (NER)

Conference on Computational Natural Language Learning (CoNLL) Corpus

Dataset Name	Year	Language	Source Type	Data Source
CoNLL'02	2002	Dutch	News wire Articles	Belgian newspaper "De Morgen"
		Spanish	News wire Articles	Spanish EFE News Agency
CoNLL'03	2003	English	News wire Articles	Reuters Corpus
		German	News wire Articles	Frankfurter Rundschau

Named Entity Recognition (NER) OntoNotes

Dataset Name	Year	Source Type	Language	Data Source
OntoNotes 1.0	2007	Newswire Articles	English	Wall Street Journal
	2007	Newswire Articles	Mandarin Chinese	Xinhua News Agency and Sinorama Magazine
OntoNotes 2.0	2008	Broadcast News	English	VoA, Public Radio International, NBC, CNN and ABC
	2008	Broadcast News	Mandarin Chinese	VoA, China Television System, China Broadcasting System, China Central TV, and China National Radio
OntoNotes 3.0	2009	Broadcast Conversation	English	Phoenix TV and China Central TV
	2009	Broadcast Conversation	Mandarin Chinese or Chinese	Phoenix TV and China Central TV
	2009	Newswire Articles	Standard Arabic or Arabic	An-Nahar
OntoNotes 4.0	2011	Weblogs, Newsgroups	English	English P2.5
	2011	Weblogs, Newsgroups	Mandarin Chinese or Chinese	Dev09, P2.5
	2011	Newswire Articles	Standard Arabic or Arabic	An-Nahar
OntoNotes 5.0	2013	Telephone, Pivot	English	English CallHome, New Testament, Old Testament
	2013	Telephone	Mandarin Chinese or Chinese	Chinese CallHome
	2013	Newswire Articles	Arabic	An-Nahar

Named Entity Recognition (NER)

Other Datasets

Dataset Name	Language	Data Source
MET [9]	Spanish, Japanese	MUC-6 dataset
IJCNLP [10]	Telugu, Bengali, Urdu, Hindi, Oriya	History of India including places and festivals
KPU-NE [11]	Urdu	Fifteen various sources including Education, Health, Science, Novels
Weibo [12]	Chinese	1,890 messages from social service provider “Weibo” with four entities GPE, person, location, and organization
Evalita	Italian	Tweets
		525 News stories taken from “L’Adige”
IREX	Japanese	Mainichi Newspaper
Mongolian [13]	Mongolian	33,209 sentences from news website

Named Entity Recognition (NER) and Relation Extraction (RE)

Study Type	Pre 2000	2001–2005		2006–2010		2011–2015		Post 2015	
	NER	NER	RE	NER	RE	NER	RE	NER	RE
Rule-based	0	0	0	1	2	0	0	1	0
Supervised	2	3	4	4	2	2	1	4	0
Semi Supervised	1	1	3	0	5	2	4	0	0
Distant Supervised	0	0	0	0	2	0	3	0	0
Unsupervised	0	1	2	1	2	1	1	1	0
Deep Learning	0	0	0	1	0	4	2	18	10
Joint Modeling	0	0	0	1	3	0	2	0	2
Transfer Learning	0	0	0	0	0	1	0	10	2
Survey	0	0	0	1	1	4	1	4	4
Total	3	5	9	9	17	14	14	38	18

Named Entity Recognition (NER)

	Technique ¹	Features/ Properties				Typ ²	Results			Lang.	Dataset
		E	W	C	O		P	R	F		
[21]	HMM, MEMM	-	Y	Y		HR				English	CONLL
										German	CONLL
[22]	Semi-CRF, JM	Y	Y		Brown Clusters, Wiki	HR	91.5	91.4	91.2	English	CONLL
[26]	US	Y			Heuristics		Low	High			MUC-7
[27]	MLP		Y		Sliding Window	HR	87.41	86.15	86.76	English	Commercial offers
							85.57	86.22	85.95		Seminar Announ.
[28]	MLP	Y			Skip-gram	HR			90.9	English	CONLL
									82.3		OntoNotes
[29]	RNN		Y	Y	Language Model	HR			91.93	English	CONLL
[30]	Bi-LSTM		Y	Y	Language Model	HR			92.22	English	CONLL
[32]	Neuro-CRF		Y			HR			89.62	English	CONLL
[33]	Neuro-CRF		Y	Y	Bi-LSTM	HR				English	CONLL
[54]	Neuro-CRF		Y	Y	Bi-LSTM	HR	Multiple languages are used.				
[34]	Neuro-CRF		Y		CNN, Iterated Dilation	HR			90.65	English	CONLL
									84.53		OntoNotes5
[35]	Neuro-CRF		Y		Memory Network				89.5	English	CONLL
[42]	HMM	Y	Y	Y	Lexicalized HMM	OTH				Chinese	Multiple Chinese Datasets
[11]	MLP	-	Y	-	Context Window	OTH	81.05	87.54	84.17	Urdu	KPU-NE
[47]	SS				TBL	OTH	76.45	99.20	86.36	Filipino	Asian Hist. Ref.
[48]	SS		Y	Y	Bootstrapping, linguistic rules	OTH	73.03	71.62	72.31	Dutch	CONLL
							78.19	76.14	77.15	Spanish	CONLL

Named Entity Recognition (NER)

	Technique ¹	Features/ Properties				Typ ²	Results			Lang.	Dataset
		E	W	C	O		P	R	F		
[49]	SS	Y	Y		Iterative	OTH				Indon- esian	75 Wikipedia Articles
[74]	RNN	Y	Y		Early Stopping, Weight Decay		85.69	80.10	82.81	Italian	Evalita (Tweets and News)
[50]	DNN		Y	Y	Bi-GRU, AdaGrad	OTH			89.92	Czech	News
[52]	DNN		Y	Y	Co-training	OTH			94.56	Vietnam	VLSP
[53]	Neuro-CRF		Y	Y	LSTM, GRU, SCRN	OTH			90.89	Korean	ETRI
[72]	Heuristic	D	-	-		DOM	99.57	93.75	96.52	English	Dietary Recom.
[73]	CRF	D	Y			DOM	67.81	52.52	58.46	English	Micropost Twitter
[87]	US				Phrase Chunking	DOM			15.2	English	GENIA
									26.5		Pittsburgh
[74]	RNN	Y	Y		Early Stopping, Weight Decay	DOM	85.69	80.10	82.81	Italian	Evalita
[88]	LSTM		Y	Y		DOM	82.70	86.70	84.60	English	Pubmed Abstracts
[75]	LSTM	Y	Y	Y	Cross domain learning	DOM			59.78	Chinese	Social Media
[79]	CNN		Y		One vs rest approach	DOM			88.64	Chinese	Discharge Summ.
									91.13		Progress Note
[80]	Neuro-CRF				Document level features		87.38	87.38	87.38	Chinese	Marriage Judge.
							94.49	88.60	91.45		Contract Judge.

Named Entity Recognition (NER)

	Technique ¹	Features/ Properties				Typ ²	Results			Lang.	Dataset
		E	W	C	O		P	R	F		
[38]	HMM	-	Y	-	-	MUL	96.00	93.00	94.47	English	MUC-6
									90.00	Spanish	MET-1
[40]	MEMM	Y	Y		Reference Resolution	MUL			90.25	English	MUC-7
									83.80	Japanese	MET-2
									77.37	Japanese	IREX
[41].	CRF	Y	Y						84.04	English	CONLL
									68.11	German	CONLL
study [43]	CLM	Y	Y	Y	Language Models, CogCompNLP	MUL	Performance at par with recent DL frameworks			Tagalog, Somali, Hindi, Farsi, Bengali, Arabic, Amharic and English	
[61]	Neuro-CRF	Y	Y		Bi-LSTM and CRF for NER, CNN for word features	MUL			70.90	Marathi	
									55.57	Bengali	
									64.27	Malayalam	
									60.25	Tamil	
[60]	TL	Y	Y		Wikipedia, Translation of lexical resources, Cross-lingual NER	MUL	Training of each model using English and one relevant language			Dutch, German, Spanish, Turkish, Bengali, Tamil, Yoruba, Uyghur	

¹SS, US, TL denote semi-supervised, unsupervised, respectively, and transfer learning.

²HR, OTH, MUL denotes high-resource, others, and multiple languages, respectively.

Relation Extraction (RE)

Study	Technique	Evaluation Metrics			Features/ Model Properties	Dataset/ Genre
		P	R	F		
[105]	MEMM			52.8	Lexical, Semantic and Syntactic	ACE'02
				55.2		ACE'03
[88]	Bi-LSTM	67.5	75.8	71.4	Stacked LSTM Model	PubMed abstracts
[99]	Heuristic	68	83	75	Conjunction, Negation	LLL'05 workshop
[101]	Heuristic	75.5	62.1	68.1	Syntactic Parser, DBPedia	Quaero News
[107]	SVM	77.2	60.7	68.0	Lexical, Semantic, Syntactic, External Lexicon	ACE'03
[109]	SVM	82.7	91.3	86.0	Kernels and voted perceptron	200 newswire and publications
[110]	SVM	70.3	26.3	38.0	Tree Kernel	ACE
[111]	SVM	76.1	68.4	72.1	Tree Kernel	ACE'03
[113]	Bootstrapping with SVM	63.2	61.5	60.3	Radial Bias Kernel	Self-annotated
[115]	CRF	73.4	56.1	63.6	Relational pattern features. Word, external	Wikipedia articles
[94]	SS				BootStrapping, Ontology	Sports and Companies web pages
[117]	SVM				Semantic Classes, Partial Pattern	TSUBAKI
[118]	SS	57.0			KBs, Tensor Decomposition	New York Times dataset [119]
[119]	Collaborative Filtering	69.0			KB, Universal Schemas	New York Times dataset

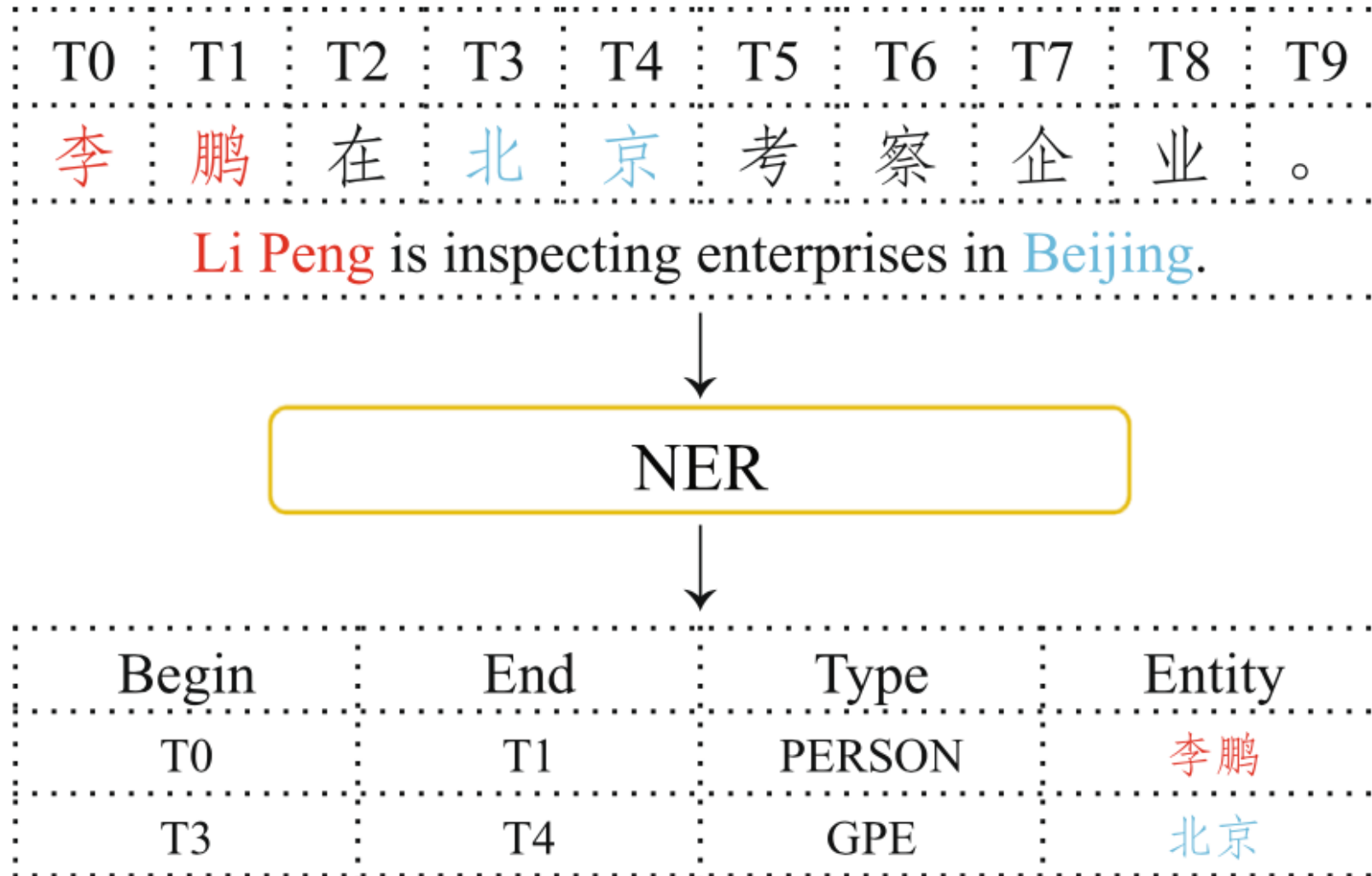
Relation Extraction (RE)

Study	Technique	Evaluation Metrics			Features/ Model Properties	Dataset/ Genre
		P	R	F		
[116]	Multi-class Logistic Regression	68.0			KBs, Lexical and Syntactical Features	Self-annotated using Mechanical Turk
[121]	DS	87.0			SampleRank, CRF, FreeBase	New York Times dataset
[123]	Logistic Regression	78.2	68.2	66.7	Freebase, EM	Wikipedia articles
[144]	LSTM				Attention Mechanism	NYT
[145]	CNN	Accuracy: 86.2 77.3			Semantic Jaccard	Wikipedia Articles New York Times
[146]	Piece-wise CNN	46.9	44.5	45.7	Word-level attention model	NYT [121]
[147]	Multi-path CNN	77.0			Word and sentence level attention model	NYT [121]
[154]	Clustering	77.5	78.5	77.5	Hierarchy NER, Complete Linkage	NYT
[125]	US				Unsupervised Feature Subset Selection, K-means	
[126]	US	Accuracy: 79.5			Chunking Information, Hierarchical Clustering	News
[127]	HMM	85.1				Web-pages
[128]	US	89.7	68.4	77.6	Hierarchical Clustering	Cluewebset'09

Relation Extraction (RE)

Study	Technique	Evaluation Metrics			Features/ Model Properties	Dataset/ Genre
		P	R	F		
[138]	RNN	82.4		82.4	POS Tags, NER Tags, Wordnet Hypernyms	SemEval 2010
[139]	CNN	82.7		82.7	Wordnet	SemEval 2010
[140]	CNN	88.0		88.0	Multi-level Attention Model	SemEval 2010
[141]	RNN	79.0		79.0	Skip-gam-based Word Vectors	SemEval 2010
[142]	LSTMs	72.9	70.8	67.9	Dynamic models	CONLL'04
[129]	Viterbi	54.0	68.4	58.14	Inferencing	TREC documents
[130]	Joint Model	90.1 73.0	91.8 62.7	91.3 66.0	POS Tags, Context Words, Hybrid Model including SVM, CYK-Parsing	TREC documents [129]
[155]	Joint Model	94.0 76.0			Graph	New York Times data
[131]	Joint Model	93.4 72.6	93.4 64.3	93.4 68.2	BootStrapping with Markov Models and CRF, Joint Model	Wikipedia
[148]	Joint Model	83.5 64.7	76.2 38.5	79.7 48.3	Casing, Gazetteer, Relation Features, Perceptron	ACE'04
		85.2 68.9	76.9 41.9	80.8 52.1		ACE'05
[132]	Joint Model	92.4 83.7	92.4 59.9	92.4 69.8	History Info., Structured Learning	TREC documents [129]
[149]	Joint Model	80.8 48.7	82.9 48.1	81.8 48.4	Bi-directional LSTM	ACE'04
		82.9 57.2	83.9 54.0	83.4 55.6		ACE'05
[143]	LSTM, Capsule Networks	30.8	63.7	41.6	Attention re-routing, position embedding	NYT
				84.5		SemEval-2010
[150]	Transfer Learning				Knowledge bases	Wiki-KBP NYT

Chinese Named Entity Recognition (CNER)



An illustration of NER task. The sample sentence is from People's daily dataset, and GPE means Geo-Political Entity.

Named Entity Recognition (NER)

Language	Ref.	Year	Topic
Universal	[1]	2007	A survey of named entity recognition and classification
Universal	[2]	2008	Named entity recognition approaches
Universal	[3]	2013	Techniques for named entity recognition: a survey
Universal	[4]	2018	An overview of named entity recognition
Universal	[5]	2018	Recent named entity recognition and classification techniques: a systematic review
Universal	[6]	2019	A survey on named entity recognition
Universal	[7]	2020	A survey on deep learning for named entity recognition
Universal	[8]	2020	A survey of named-entity recognition methods for food information extraction

Named Entity Recognition (NER)

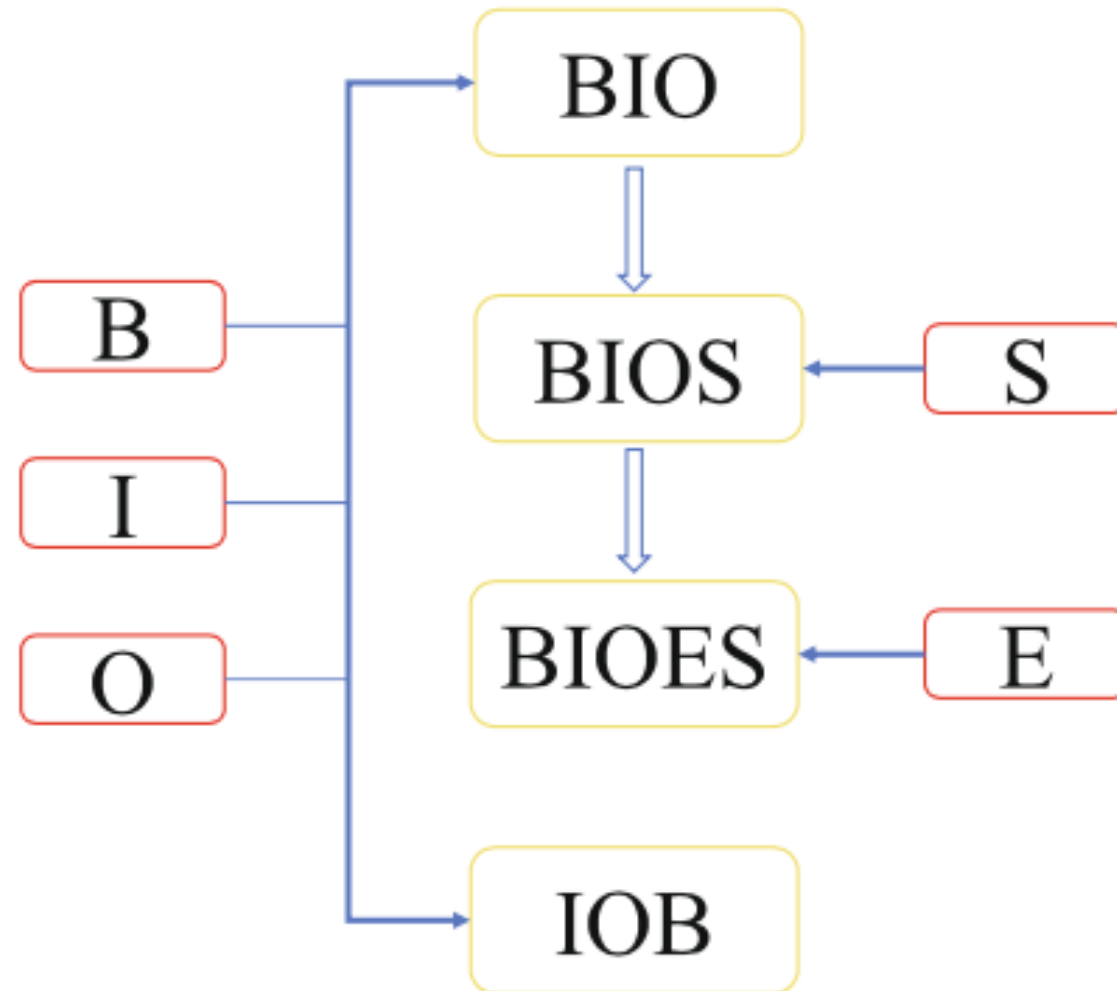
Language	Ref.	Year	Topic
Arabic	[13]	2017	A comparative review of machine learning for Arabic named entity recognition
Arabic	[14]	2019	Arabic named entity recognition using deep learning approach
Arabic	[15]	2019	Arabic named entity recognition: What works and what's next
Indian	[16]	2010	A survey of named entity recognition in English and other Indian languages
Indian	[17]	2011	A survey on named entity recognition in Indian languages with particular reference to Telugu
Indian(Assamese)	[18]	2014	A survey of named entity recognition in Assamese and other Indian languages
Indian(Hindi)	[19]	2016	Survey of named entity recognition systems with respect to Indian and foreign languages
Indian	[20]	2017	Survey of named entity recognition techniques for various Indian regional languages
Indian(Hindi)	[21]	2019	Named entity recognition for Hindi language: A survey
Indian	[22]	2019	Named entity recognition: A survey for Indian languages
Indian(Hindi)	[23]	2020	A survey on various methods used in named entity recognition for hindi language
English	[24]	2013	Named entity recognition in english using hidden markov model
Marathi	[25]	2016	Issues and Challenges in Marathi Named Entity Recognition
Turkish	[26]	2017	Named entity recognition in Turkish: Approaches and issues
Spanish	[27]	2020	Named entity recognition in Spanish biomedical literature: Short review and bert model

Public datasets of Chinese NER

#Tags: the number of entity types

Corpus	#Tags	Entity types	URL
WEIBO	4	Person, Location, Organization and Geo-political	https://github.com/hltcoe/golden-horse
MSRA	3	Person, Location, Organization	https://github.com/InsaneLife/ChineseNLPCorpus/tree/master/NER/MSRA
People's Daily	4	Person, Organization, Geo-political, Date	https://github.com/GuocaiL/nlp_corpus/tree/main/open_ner_data/people_daily
bosonNLP	6	Person, Location, Organization, Company, Product, Time	https://github.com/InsaneLife/ChineseNLPCorpus/tree/master/NER/boson
RESUME	8	Person, Location, Organization, Country, Education, Profession, Race, Title	https://github.com/GuocaiL/nlp_corpus/tree/main/open_ner_data/ResumeNER
OntoNotes Release 5.0	18	Person, NORP, Facility, Organization, GPE, Location, Product, Event, Work of art, Law, Language, Date, Time, Percent, Money, Quantity, Ordinal, Cardinal	https://doi.org/10.35111/xmhb-2b84
CLUENER 2020	10	Address, Book, Company, Game, Government, Movie, Name, Organization, Position, Scene	https://github.com/CLUEbenchmark/CLUENER2020

Evolution of four commonly used tag schemes



Named Entity Recognition (NER)

李 鹏 在 北 京 考 察 企 业 。

Li Peng is inspecting enterprises in Beijing.

Regular expression

Last name(李,王,...)
+ First name(鹏,...)

Dictionary

[张三(Zhangsan),
李鹏(Lipeng),
王五(Wangwu)...]

An illustration of rule-based methods.

A person's name is matched by a regular expression and a dictionary.

Named Entity Recognition (NER)

Lexicon	朝阳 (morning sun)	明朝 (Ming Dynasty)	朝鲜半岛 (Korean Peninsula)	朝夕 (morning and evening)
Glyph	 Oracle Bone Script	 Bronze Script	 Clerical Script	 Regular Script
Radical	十 (ten)	日 (sun)	十 (ten)	月 (moon)
Stroker	一	丿 一一	一	丿 丿 一一

The illustration of some external information of the character “朝”

Named Entity Recognition (NER)

Improvement brought by BERT in different works

Work	Dataset	Model	F1(%)	Improvement(%)	Year
[63]	MSRA	Word2Vec + radical + BGRU-CRF	90.45	4.97	2019
		BERT + radical + BGRU-CRF	95.42		
[74]	MSRA	PLTE	93.26	1.27	2020
		PLTE[BERT]	94.53		
	Ontonotes	PLTE	74.60	6.00	
		PLTE[BERT]	80.60		
	Weibo	PLTE	55.15	14.08	
		PLTE[BERT]	69.23		
[75]	MSRA	SoftLexicon(LSTM)	93.66	1.76	2020
		SoftLexicon(LSTM)+BERT	95.42		
	Ontonotes	SoftLexicon(LSTM)	75.64	7.17	
		SoftLexicon(LSTM)+BERT	82.81		
	Weibo	SoftLexicon(LSTM)	61.42	9.08	
		SoftLexicon(LSTM)+BERT	70.50		
[76]	CCKS2018	Word2Vec + CRF	69.01	21.53	2020
		BERT + CRF	90.54		
		Word2Vec + BiLSTM-CRF	75.60	15.83	
		BERT + BiLSTM-CRF	91.43		

Named Entity Recognition (NER)

The effect of POS and radical information

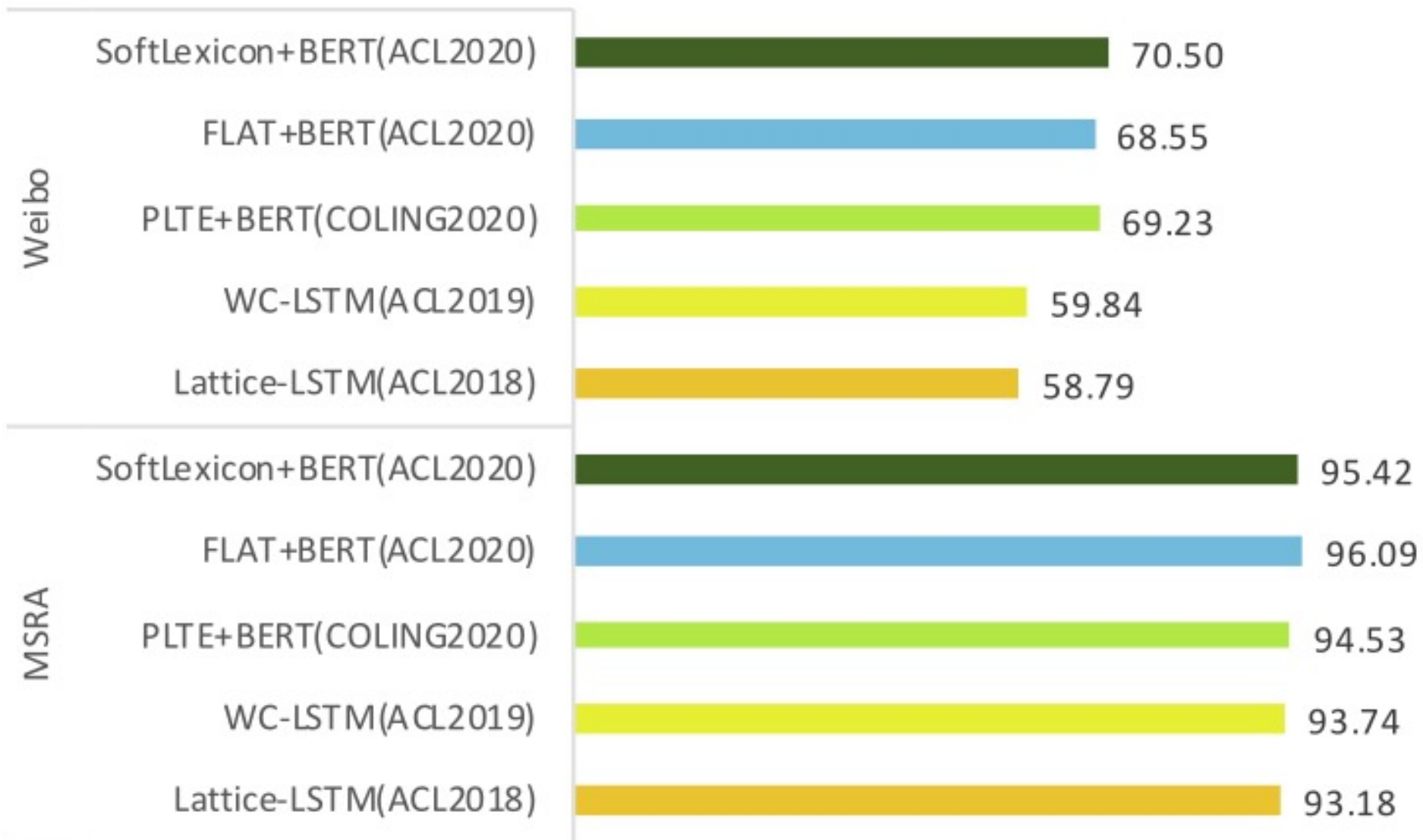
Work	Dataset	Model	F1(%)	Improvement(%)	Year
[55]	MSRA	random + dropout	88.91	0.53	2016
		random + radical + dropout	89.44		
[58]	CCKS2018	LSTM-CRF	67.32	11.62	2019
		POS + LSTM-CRF	78.94		
		SM-LSTM-CRF	69.91	10.16	
		POS + SM-LSTM-CRF	80.07		
[60]	CCKS2017	BILSTM-CRF	88.78	Baseline	2019
		BILSTM-CRF + radical	89.64	0.86	
		BILSTM-CRF + POS	89.06	0.28	
		BILSTM-CRF + radical + POS	90.12	1.34	
		Att-BILSTM-CRF	90.11	Baseline	
		Att-BILSTM-CRF + radical	90.96	0.85	
		Att-BILSTM-CRF + POS	90.81	0.70	
		Att-BILSTM-CRF + radical + POS	91.35	1.24	
[61]	CCKS2017	CRF	85.14	1.87	2019
		POS + CRF	87.01		
		BILSTM-CRF	89.66	-0.11	
		POS + BILSTM-CRF	89.55		
	CCKS2018	CRF	82.49	0.93	
		POS + CRF	83.42		
		BILSTM-CRF	84.13	-0.17	
		POS + BILSTM-CRF	83.96		

Named Entity Recognition (NER)

Improvement brought by Glyph information

Work	Dataset	Model	F1(%)	Improvement(%)	Year
[81]	MSRA	BERT	94.80	0.74	2019
		BERT + Glyce	95.54		
		Lattice-LSTM	93.18	0.71	
		Lattice-LSTM + Glyce	93.89		
[57]	MSRA	BILSTM-CRF	89.94	1.14	2019
		BILSTM-CRF + glyph embeddings	91.08		
[83]	MSRA	BERT + BILSTM-CRF	95.30	1.19	2019
		BERT + BILSTM-CRF + GLYNN	96.49		

Named Entity Recognition (NER)



Named Entity Recognition (NER)

The illustration of representations of the character “朝”

	Pre-trained character embeddings		External Information					
	Lookup tables	Pre-trained language models	POS	Radical Information	Stroke Information	Glyph Information	Lexicon Information	Chinese Word Segmentation
Illustrations	Word2vec, Glove, FastText, etc.	BERT, ELMo, NEZHA, etc.	Noun	十日十月	一丨丨フ一 一一丨フ 一一	朝	朝阳, 明朝, 朝夕, 朝鲜	明朝/的/皇帝
methods	Static embeddings	Contextual embeddings	embeddings	embeddings and RNN	RNN	CNN	Lattice	CWS tools

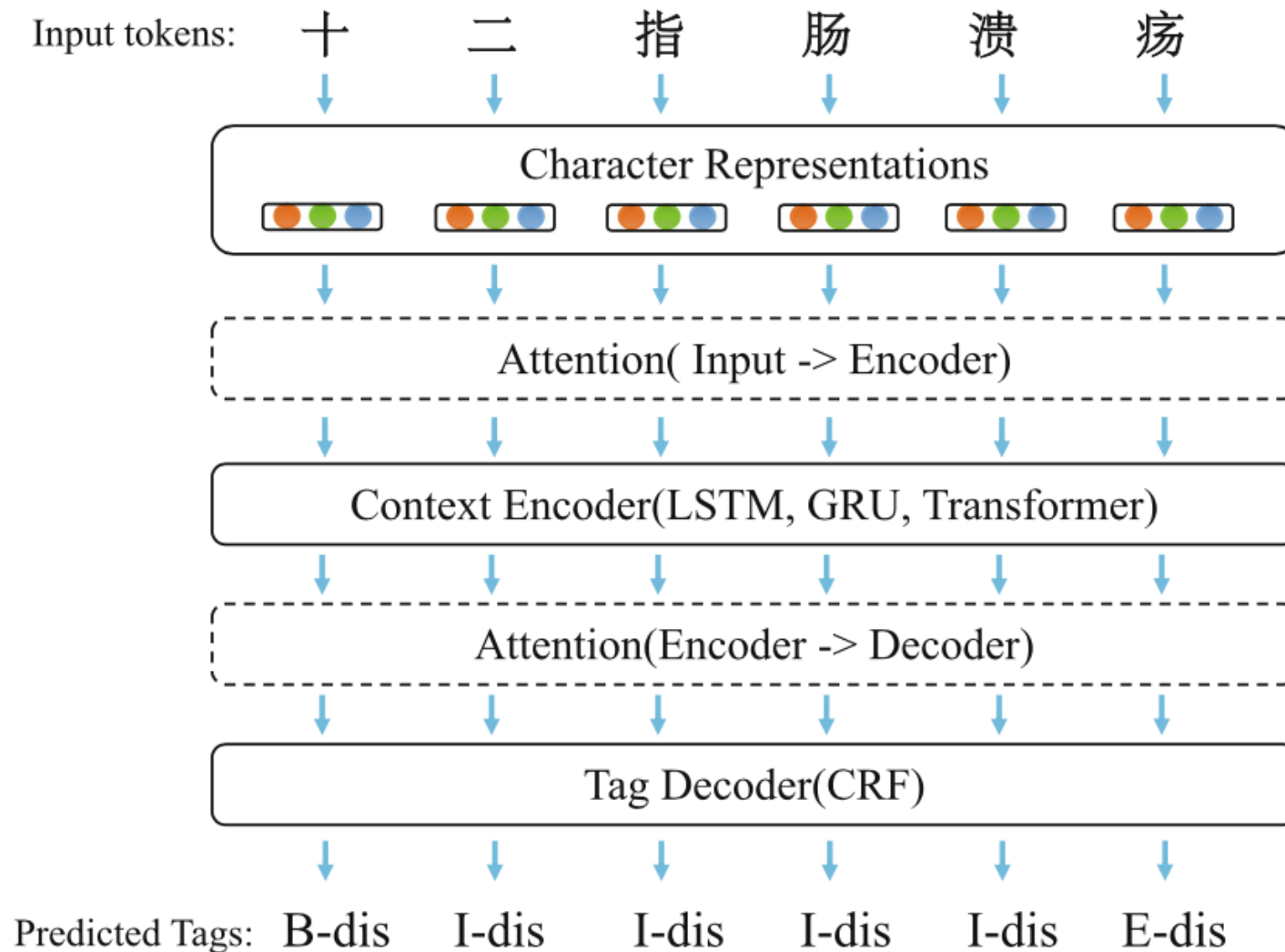
Chinese Named Entity Recognition (CNER)

Work	Character representation		Attention		Attention		Performance (F1-score)	Year
	Character embeddings	External Information	Input -> Encoder	Context Encoder	Encoder -> Decoder	Tag Decoder		
[55]	Word2vec	Radical	✓	LSTM	✓	CRF	MSRA:89.78%	2016
[58]		POS		LSTM		CRF	MSRA:90.95%	2019
[59]	✓			LSTM		CRF	CCKS2018:78.94%	2019
[60]	✓	POS, Radical		LSTM	✓	CRF	CCKS2018:80.07%	2019
[61]	✓	POS, Dictionary		LSTM	✓	CRF	CCKS2018:86.68%	2019
[80]	Word2vec	CWS, Radical, Lexicon, Stroker		LSTM		CRF	CCKS2018:87.26%	2019
[76]	Word2vec			O		CRF	CCKS2017:90.12%	2019
	BERT						CCKS2017:91.35%	2019
	ERNIE						CCKS2017:90.48%	2019
	ALBERT						CCKS2018:86.11%	2019
	NEZHA						CCKS2017:91.75%	2020
	Word2vec			LSTM			CCKS2018:90.05%	2020
	BERT						CCKS2018:69.01%	2020
	ERNIE						CCKS2018:90.54%	2020
	ALBERT						CCKS2018:93.37%	2020
	NEZHA						CCKS2018:87.68%	2020
[56]	Sogou news	Radical		LSTM		CRF	CCKS2018:93.58%	2019
	Word2vec						CCKS2018:75.60%	2019
[62]	✓	Position, segmentation		GRU	✓	CRF	CCKS2018:91.43%	2019
			Concolution - attention		✓		CCKS2018:93.11%	2019
							CCKS2018:90.12%	2019
							CCKS2018:95.08%	2019
							Peoples'Daily:92.06%	2019
							Peoples'Daily:94.37%	2019
							WEIBO:53.80% MSRA:90.32%	2019
							WEIBO:55.91% MSRA:92.34%	2019
							WEIBO:59.31% MSRA:92.97%	2019

Chinese Named Entity Recognition (CNER)

Work	Character representation		Attention		Attention		Performance (F1-score)	Year
	Character embeddings	External Information	Input -> Encoder	Context Encoder	Encoder -> Decoder	Tag Decoder		
[63]	Word2vec BERT	Radical		GRU		CRF	MSRA:90.45% MSRA:95.42%	2019
[77]	Conv-GRU Embedding	Word, Radical		GRU		CRF	WEIBO:68.93% MSRA:91.45%	2019
[88]	✓	Dictionary		LSTM		CRF	CCKS2017:91.24%	2019
[64]	✓	Lexicon, Word		LSTM		CRF	WEIBO:63.09% MSRA:93.47%	2019
[65]	✓	Word, Position		LSTM	✓	CRF	WEIBO:59.5% MSRA:92.99%	2020
[81]	BERT	Glyph		Transformer		CRF	WEIBO:67.60% MSRA:95.54%	2019
[57]	Wikipedia GloVe	Glyph		LSTM		CRF	MSRA:91.11%	2019
[82]	BERT	Radical, Glyph		LSTM		CRF	WEIBO:70.01% MSRA:95.51%	2020
[83]	BERT	Glyph		LSTM		CRF	WEIBO:71.81% MSRA:96.49%	2019
[66]	✓	Radical, Word	✓	GRU		CRF	WEIBO:71.86% MSRA:92.71%	2020
[67]	✓	Adapted GGNN Gazetteers		LSTM		CRF	WEIBO:59.5% MSRA:94.4%	2020
[85]	✓	Lexicon		Lattice-LSTM		CRF	WEIBO:58.79% MSRA:93.18%	2018
[86]	✓	Lexicon		WC-LSTM		CRF	WEIBO:59.84% MSRA:93.36%	2019
[74]	✓	Lexicon		PLTE		CRF	WEIBO:55.15% MSRA:93.26%	2019
	BERT						WEIBO:69.23% MSRA:94.53%	
[87]	BERT			MLP		CRF	WEIBO:68.20% MSRA:94.95%	2020
	✓	Lexicon		FLAT			WEIBO:63.42% MSRA:94.35%	
	BERT						WEIBO:68.55% MSRA:96.09%	
[75]	✓	SoftLexicon		LSTM		CRF	WEIBO:61.42% MSRA:93.66%	2020
	BERT						WEIBO:70.50% MSRA:95.42%	
[99]	BERT	Lexicon, radical		Transformer	✓	CRF	WEIBO:70.43% MSRA:96.24%	2021

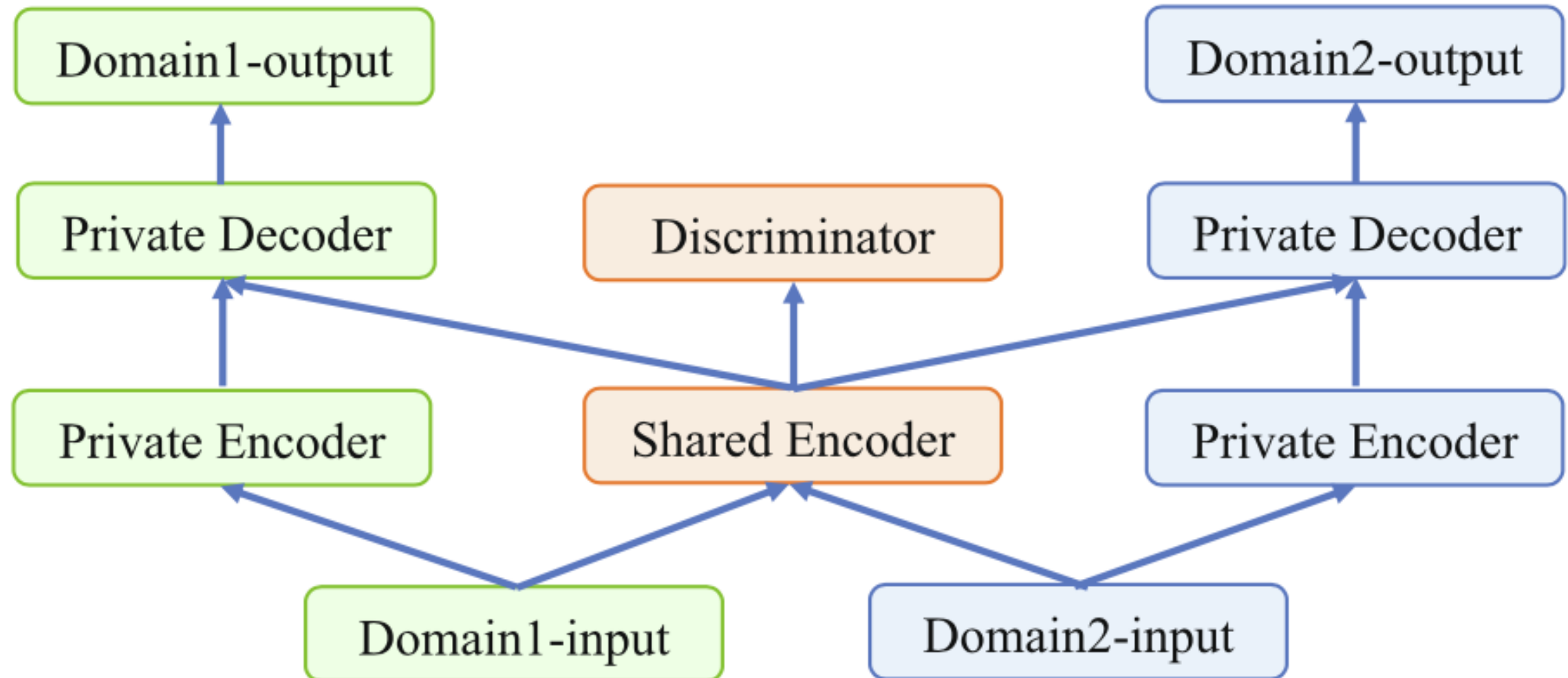
Named Entity Recognition (NER)



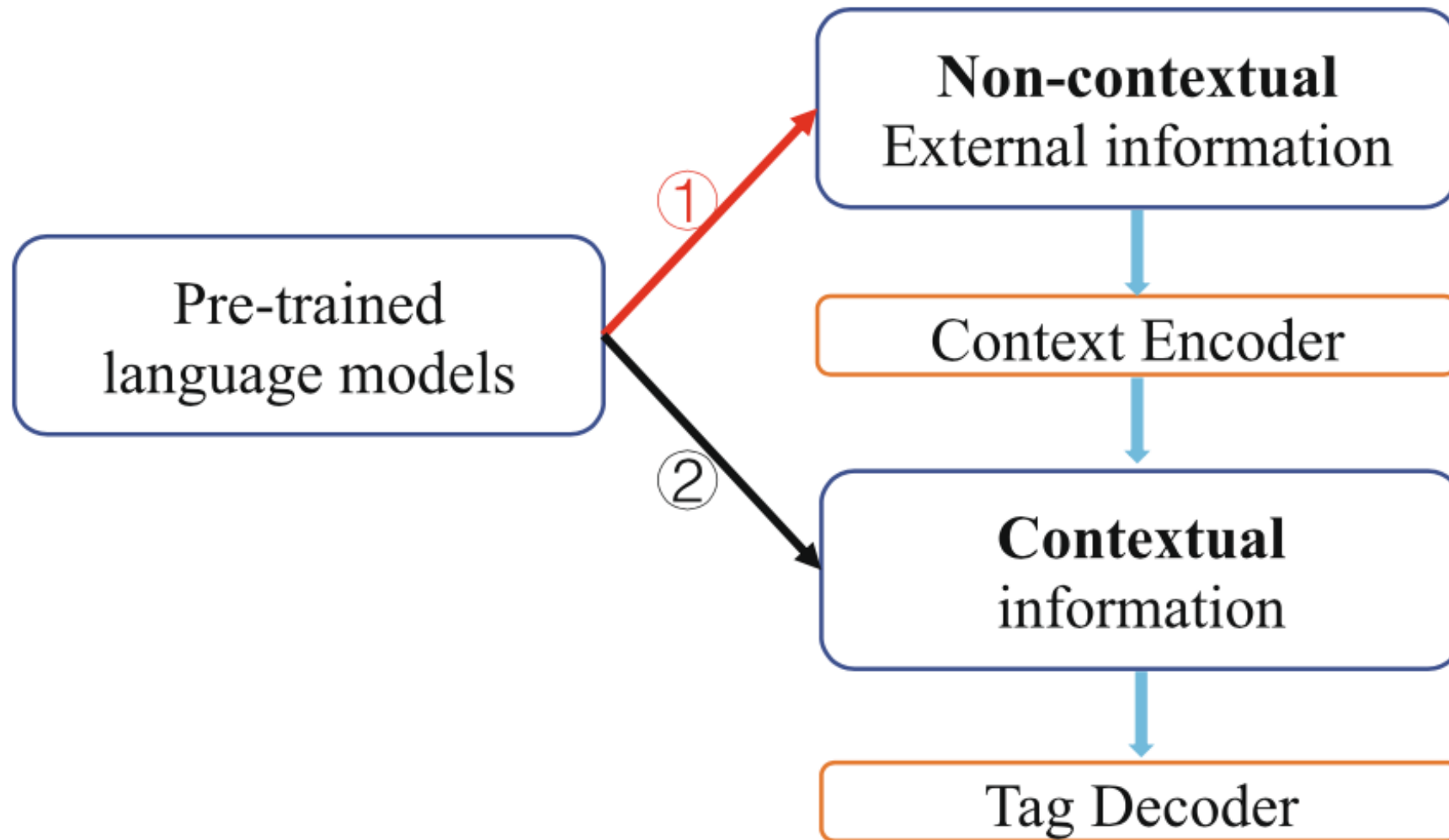
Chinese Named Entity Recognition (CNER) using attention modules

Work	Dataset	Model	F1(%)	Improvement(%)
[58]	CCKS2018	LSTM-CRF	67.32	2.59
		SM-LSTM-CRF	69.91	
		POS + LSTM-CRF	78.94	
		POS + SM-LSTM-CRF	80.07	
[60]	CCKS2017	BILSTM-CRF	88.78	1.33
		Att-BILSTM-CRF	90.11	
		BILSTM-CRF + radical	89.64	1.32
		Att-BILSTM-CRF + radical	90.96	
		BILSTM-CRF + POS	89.06	1.75
		Att-BILSTM-CRF + POS	90.81	
		BILSTM-CRF + radical + POS	90.12	1.23
		Att-BILSTM-CRF + radical + POS	91.35	
[59]	CCKS2018	BILSTM-CRF	86.68	0.58
		Attention-BILSTM-CRF	87.26	
		BILSTM-CRF + dictionary	87.71	0.58
		Attention-BILSTM-CRF + dictionary	88.29	
[56]	CCKS2018	char	86.09	3.17
		char + attention	89.26	
		char + word	90.74	3.74
		char + word + attention	94.48	

Schematic diagram of cross-domain adversarial transfer learning



Two ways of concatenating the representations of pre-trained language models and external information



Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb

File Edit View Insert Runtime Tools Help Last edited on May 13

Table of contents

- Text Analytics and Natural Language Processing (NLP)
- Python for Natural Language Processing**
 - spaCy Chinese Model
 - Open Chinese Convert (OpenCC, 開放中文轉換)
 - Jieba 結巴中文分詞
 - Natural Language Toolkit (NLTK)
 - Stanza: A Python NLP Library for Many Human Languages
- Text Processing and Understanding
 - NLTK (Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit)
 - NLP Zero to Hero
 - Natural Language Processing - Tokenization (NLP Zero to Hero, part 1)
 - Natural Language Processing - Sequencing - Turning sentence into data (NLP Zero to Hero, part 2)
 - Natural Language Processing - Training a model to recognize sentiment in text (NLP Zero to Hero, part 3)
 - Keras preprocessing text
 - JSON File

```
1 text = "Steve Jobs and Steve Wozniak incorporated Apple Computer on January 3, 1977, in Cupertino, California."
2 doc = nlp(text)
3 displacy.render(doc, style="ent", jupyter=True)
```

Steve Jobs PERSON and Steve Wozniak PERSON incorporated Apple Computer ORG on January 3, 1977 DATE , in Cupertino GPE , California GPE .

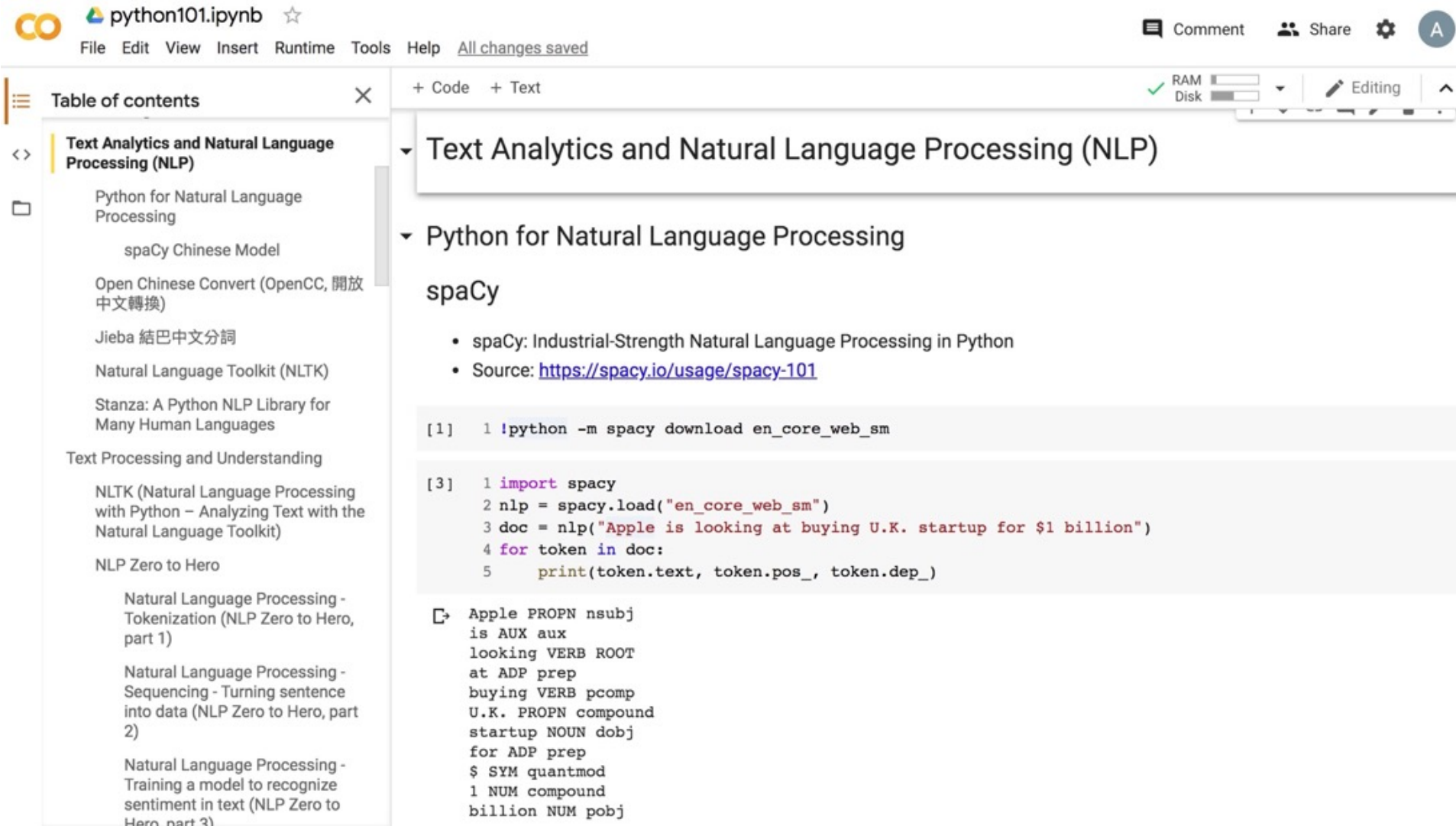
```
[ ] 1 import spacy
2 nlp = spacy.load("en_core_web_sm")
3 doc = nlp("Stanford University is located in California. It is a great university.")
4 import pandas as pd
5 cols = ("text", "lemma", "pos", "tag", "pos_explain", "stopword")
6 rows = []
7 for t in doc:
8     row = [t.text, t.lemma_, t.pos_, t.tag_, spacy.explain(t.pos_), t.is_stop]
9     rows.append(row)
10 df = pd.DataFrame(rows, columns=cols)
11 df
```

	text	lemma	pos	tag	pos_explain	stopword
0	Stanford	Stanford	PROPN	NNP	proper noun	False
1	University	University	PROPN	NNP	proper noun	False
2	is	be	VERB	VBZ	verb	True
3	located	locate	VERB	VRB	verb	False
4	in	in	ADP	IN	adposition	True
5	California	California	PROPN	NNP	proper noun	False
6	.	.	PUNCT	.	punctuation	False
7	It	-PRON-	PRON	PRP	pronoun	True

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk

Editing

Table of contents

- Text Analytics and Natural Language Processing (NLP)
 - Python for Natural Language Processing
 - spaCy Chinese Model
 - Open Chinese Convert (OpenCC, 開放中文轉換)
 - Jieba 結巴中文分詞
 - Natural Language Toolkit (NLTK)
 - Stanza: A Python NLP Library for Many Human Languages
 - Text Processing and Understanding
 - NLTK (Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit)
 - NLP Zero to Hero
 - Natural Language Processing - Tokenization (NLP Zero to Hero, part 1)
 - Natural Language Processing - Sequencing - Turning sentence into data (NLP Zero to Hero, part 2)
 - Natural Language Processing - Training a model to recognize sentiment in text (NLP Zero to Hero, part 3)

Text Analytics and Natural Language Processing (NLP)

Python for Natural Language Processing

spaCy

- spaCy: Industrial-Strength Natural Language Processing in Python
- Source: <https://spacy.io/usage/spacy-101>

```
[1] 1 !python -m spacy download en_core_web_sm
```

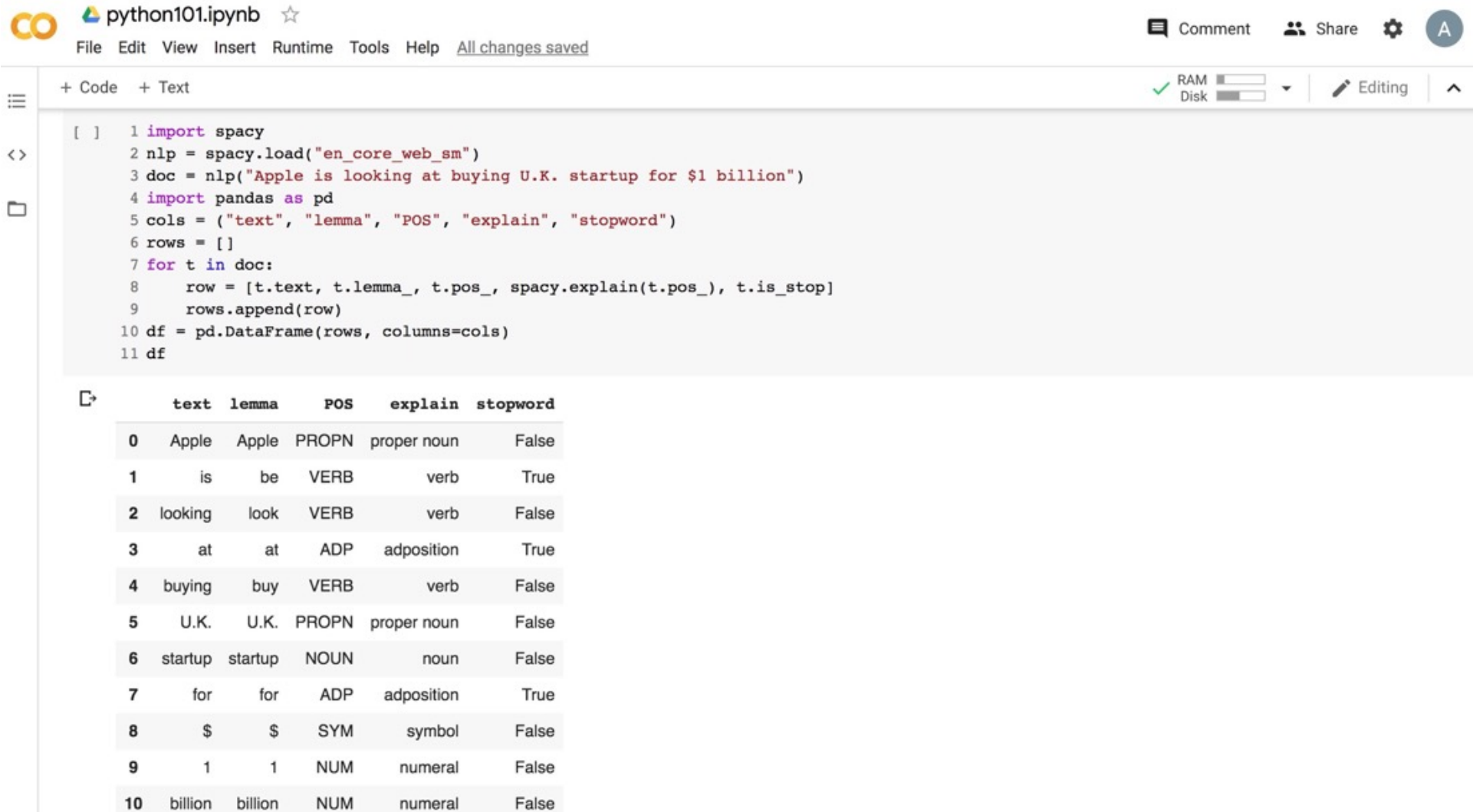
```
[3] 1 import spacy
2 nlp = spacy.load("en_core_web_sm")
3 doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
4 for token in doc:
5     print(token.text, token.pos_, token.dep_)
```

Apple PROPn nsubj
is AUX aux
looking VERB ROOT
at ADP prep
buying VERB pcomp
U.K. PROPn compound
startup NOUN dobj
for ADP prep
\$ SYM quantmod
1 NUM compound
billion NUM pobj

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



The screenshot shows a Google Colab notebook interface. At the top, the title bar says 'python101.ipynb' with a star icon. Below it, a menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help', followed by the text 'All changes saved'. On the right side of the title bar, there are icons for 'Comment', 'Share', a settings gear, and a user profile icon labeled 'A'. Below the title bar, a toolbar shows '+ Code' and '+ Text' buttons, a RAM and Disk usage indicator (both at 0%), and an 'Editing' mode button. The main area of the notebook contains a Python script that uses spaCy to process a sentence and pandas to store the results in a DataFrame. The script is as follows:

```
[ ] 1 import spacy
2 nlp = spacy.load("en_core_web_sm")
3 doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
4 import pandas as pd
5 cols = ("text", "lemma", "POS", "explain", "stopword")
6 rows = []
7 for t in doc:
8     row = [t.text, t.lemma_, t.pos_, spacy.explain(t.pos_), t.is_stop]
9     rows.append(row)
10 df = pd.DataFrame(rows, columns=cols)
11 df
```

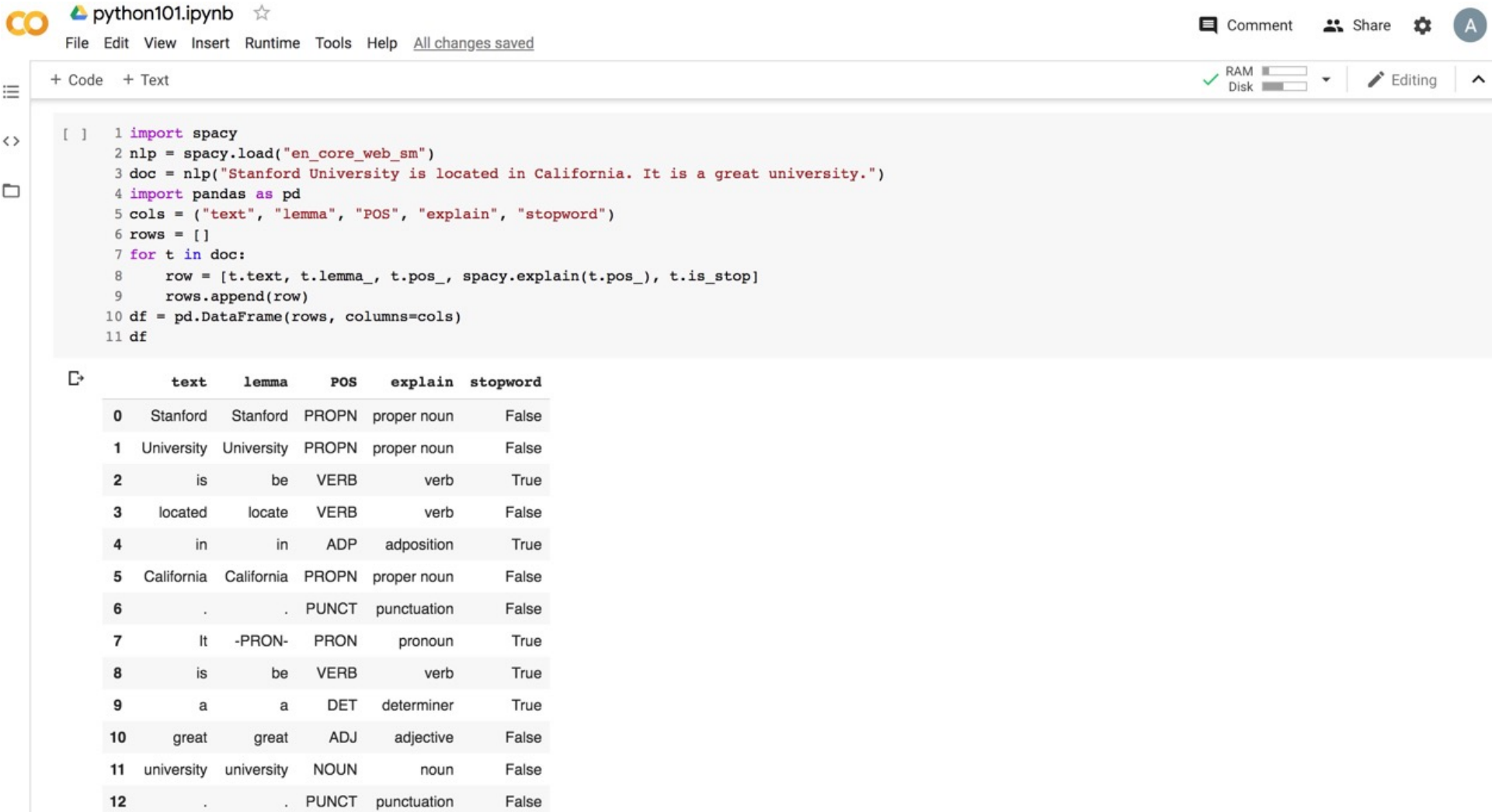
Below the code, the output is displayed as a DataFrame with 11 rows and 6 columns: 'text', 'lemma', 'POS', 'explain', and 'stopword'. The DataFrame shows the tokenization and part-of-speech tagging for the sentence "Apple is looking at buying U.K. startup for \$1 billion".

	text	lemma	POS	explain	stopword
0	Apple	Apple	PROPN	proper noun	False
1	is	be	VERB	verb	True
2	looking	look	VERB	verb	False
3	at	at	ADP	adposition	True
4	buying	buy	VERB	verb	False
5	U.K.	U.K.	PROPN	proper noun	False
6	startup	startup	NOUN	noun	False
7	for	for	ADP	adposition	True
8	\$	\$	SYM	symbol	False
9	1	1	NUM	numeral	False
10	billion	billion	NUM	numeral	False

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



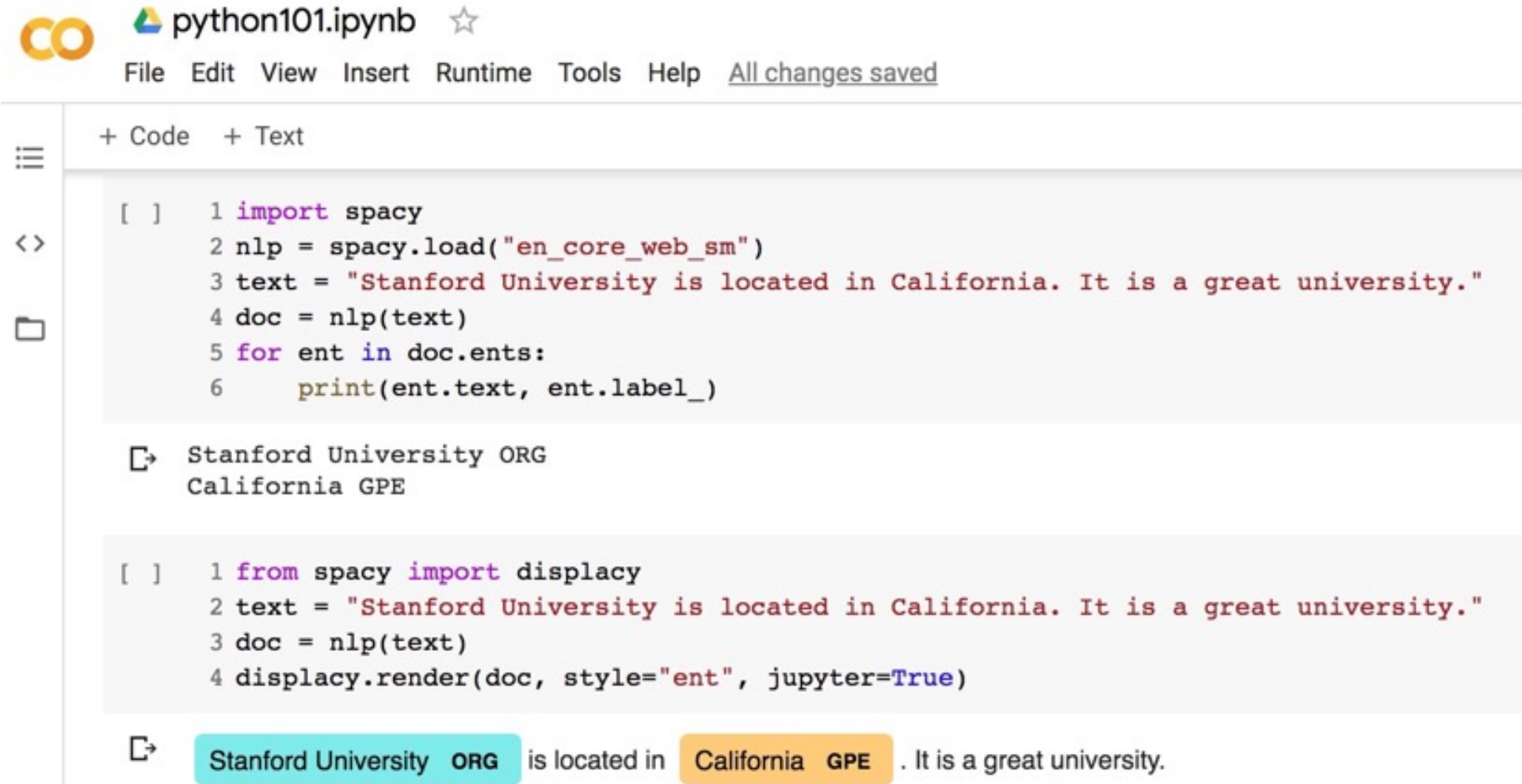
```
[ ] 1 import spacy
2 nlp = spacy.load("en_core_web_sm")
3 doc = nlp("Stanford University is located in California. It is a great university.")
4 import pandas as pd
5 cols = ("text", "lemma", "POS", "explain", "stopword")
6 rows = []
7 for t in doc:
8     row = [t.text, t.lemma_, t.pos_, spacy.explain(t.pos_), t.is_stop]
9     rows.append(row)
10 df = pd.DataFrame(rows, columns=cols)
11 df
```

	text	lemma	POS	explain	stopword
0	Stanford	Stanford	PROPN	proper noun	False
1	University	University	PROPN	proper noun	False
2	is	be	VERB	verb	True
3	located	locate	VERB	verb	False
4	in	in	ADP	adposition	True
5	California	California	PROPN	proper noun	False
6	.	.	PUNCT	punctuation	False
7	It	-PRON-	PRON	pronoun	True
8	is	be	VERB	verb	True
9	a	a	DET	determiner	True
10	great	great	ADJ	adjective	False
11	university	university	NOUN	noun	False
12	.	.	PUNCT	punctuation	False

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



The image shows a Google Colab notebook interface. At the top, there's a header with the Colab logo, the notebook name 'python101.ipynb', and a star icon. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', 'Help', and 'All changes saved'. The notebook has two code cells. The first cell contains code to import spacy, load the 'en_core_web_sm' model, and process a sentence about Stanford University. The output shows the extracted entities: 'Stanford University' (ORG) and 'California' (GPE). The second cell contains code to use the displacy library to render the entities in a visual format. The output shows the same sentence with 'Stanford University' highlighted in a light blue box labeled 'ORG' and 'California' highlighted in a light orange box labeled 'GPE'.

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[ ] 1 import spacy
    2 nlp = spacy.load("en_core_web_sm")
    3 text = "Stanford University is located in California. It is a great university."
    4 doc = nlp(text)
    5 for ent in doc.ents:
    6     print(ent.text, ent.label_)
```

Stanford University ORG
California GPE

```
[ ] 1 from spacy import displacy
    2 text = "Stanford University is located in California. It is a great university."
    3 doc = nlp(text)
    4 displacy.render(doc, style="ent", jupyter=True)
```

Stanford University ORG is located in California GPE . It is a great university.

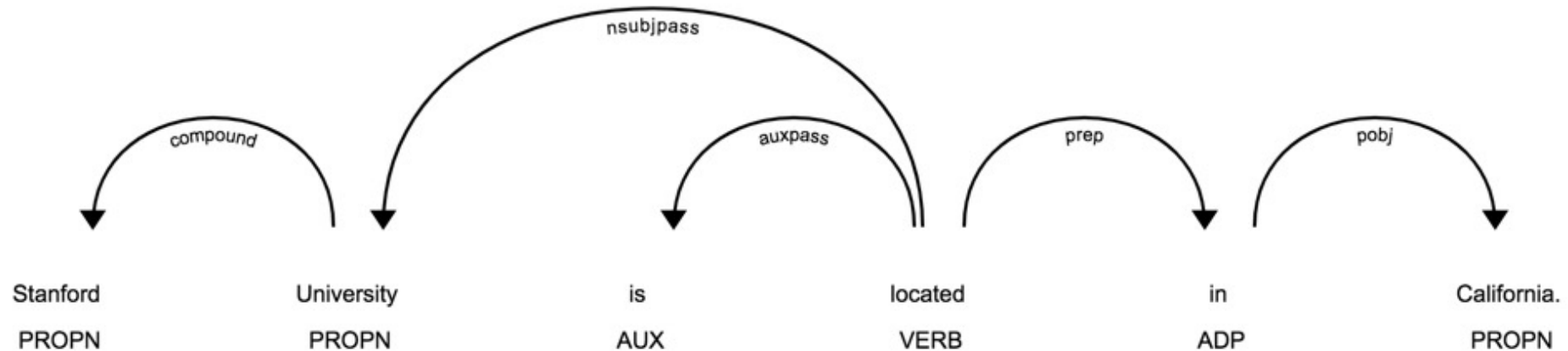
<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

```
1 from spacy import displacy
2 text = "Stanford University is located in California. It is a great university."
3 doc = nlp(text)
4 displacy.render(doc, style="ent", jupyter=True)
5 displacy.render(doc, style="dep", jupyter=True)
```

Stanford University **ORG** is located in **California GPE** . It is a great university.



<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot displays a Google Colab notebook environment. The top bar includes the Colab logo, the notebook name 'python101.ipynb', and navigation links like 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. A 'Table of contents' sidebar on the left lists various NLP topics. The main code cell contains two Python snippets. The first snippet uses spaCy to process a sentence and render it with part-of-speech tags. The second snippet imports spaCy, loads a model, processes a sentence, and creates a DataFrame to analyze the tokens. The output of the second cell shows a table with columns: text, lemma, pos, tag, pos_explain, and stopwords.

	text	lemma	pos	tag	pos_explain	stopword
0	Stanford	Stanford	PROPN	NNP	proper noun	False
1	University	University	PROPN	NNP	proper noun	False
2	is	be	VERB	VBZ	verb	True
3	located	locate	VERB	VRB	verb	False
4	in	in	ADP	IN	adposition	True
5	California	California	PROPN	NNP	proper noun	False
6	.	.	PUNCT	.	punctuation	False
7	It	-PRON-	PRON	PRP	pronoun	True

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Table of contents

- Text Classification: BBC News Articles
- Text Summarization and Topic Modeling
- Text Summarization
 - Text Summarization with Gensim
 - Text Summarization
- Topic Modeling
 - Topic Modeling with Gensim LSI model
 - Topic Modeling with Gensim LDA model
 - Topic Modeling with Scikit-learn LDA and NMF
 - Topic Modeling Visualization
- Text Similarity and Clustering
 - Text Similarity
 - Text Clustering
- Semantic Analysis and Named Entity Recognition (NER)**
 - Semantic Analysis
 - Named Entity Recognition (NER)

Semantic Analysis and Named Entity Recognition (NER)

- Source: Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress. <https://github.com/Apress/text-analytics-w-python-2e>

Semantic Analysis

```
[1] 1 import nltk
    2 from nltk.corpus import wordnet as wn
    3 import pandas as pd
    4 nltk.download('wordnet')
    5 # WordNet Synsets
    6 word = 'fruit'
    7 synsets = wn.synsets(word)
    8 print('Word:', word)
    9 print('Wordnet Synsets:', len(synsets))
   10 df = pd.DataFrame([{'Synset': synset,
   11                    'Part of Speech': synset.lexname(),
   12                    'Definition': synset.definition(),
   13                    'Lemmas': synset.lemma_names(),
   14                    'Examples': synset.examples()}
   15                    for synset in synsets])
   16 df
```

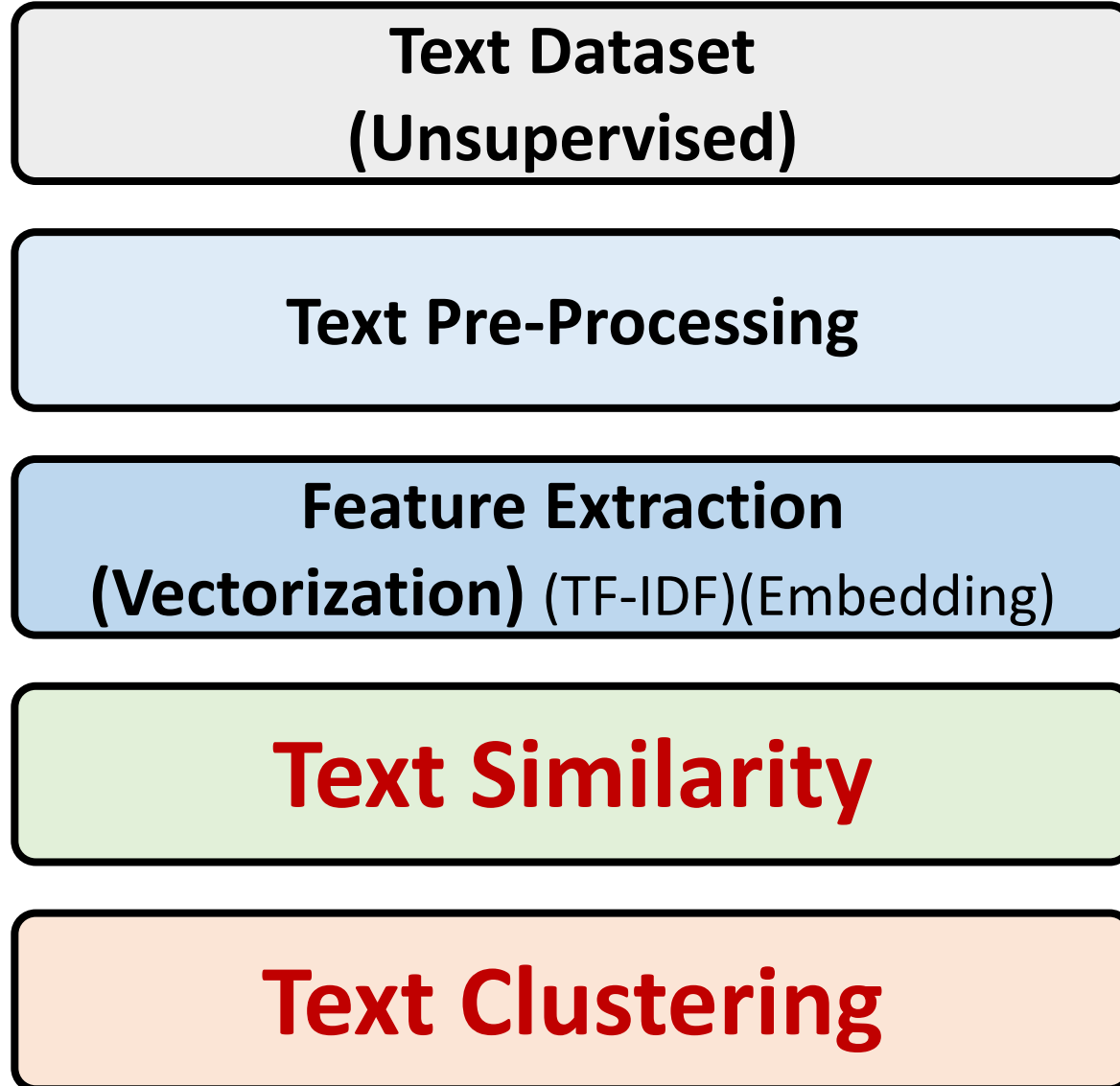
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Unzipping corpora/wordnet.zip.
Word: fruit
Wordnet Synsets: 5

	Synset	Part of Speech	Definition	Lemmas	Examples
0	Synset('fruit.n.01')	noun.plant	the ripened reproductive body of a seed plant	[fruit]	[]
1	Synset('yield.n.03')	noun.artifact	an amount of a product	[yield, fruit]	[]

<https://tinyurl.com/aintpuppython101>

Text Similarity and Clustering

Text Similarity and Clustering



Text Similarity and Clustering

- How do we measure **similarity** between terms and documents?
- How can we use **distance** measures to find the most **relevant documents**?
- How can we build a **recommender system** from **text similarity**?
- How do we **group similar documents (document clustering)**?

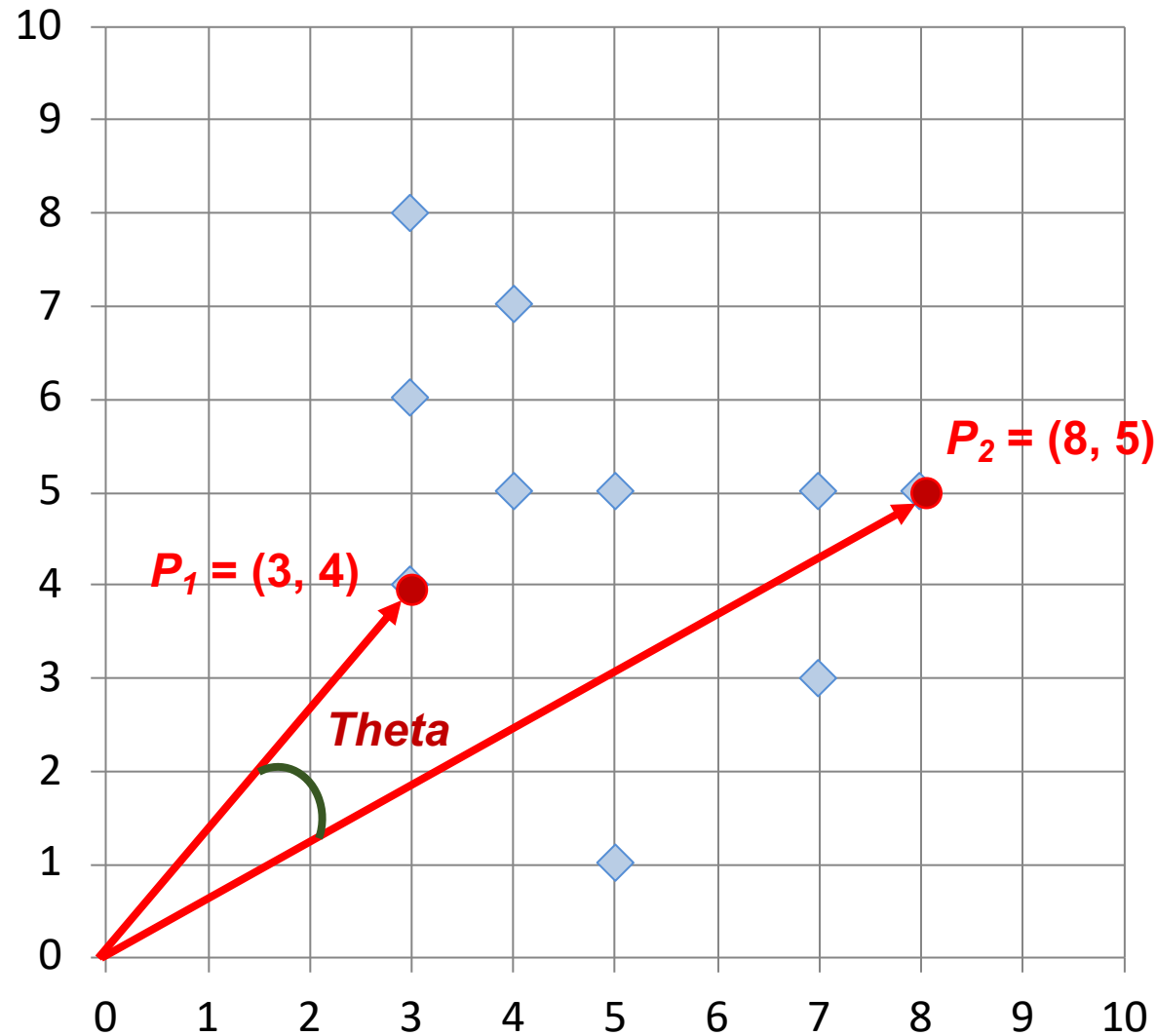
Text Similarity and Clustering

- **Information Retrieval (IR)**
- **Feature Engineering**
- **Similarity Measures**
- **Unsupervised Machine Learning Algorithms**

Text Similarity

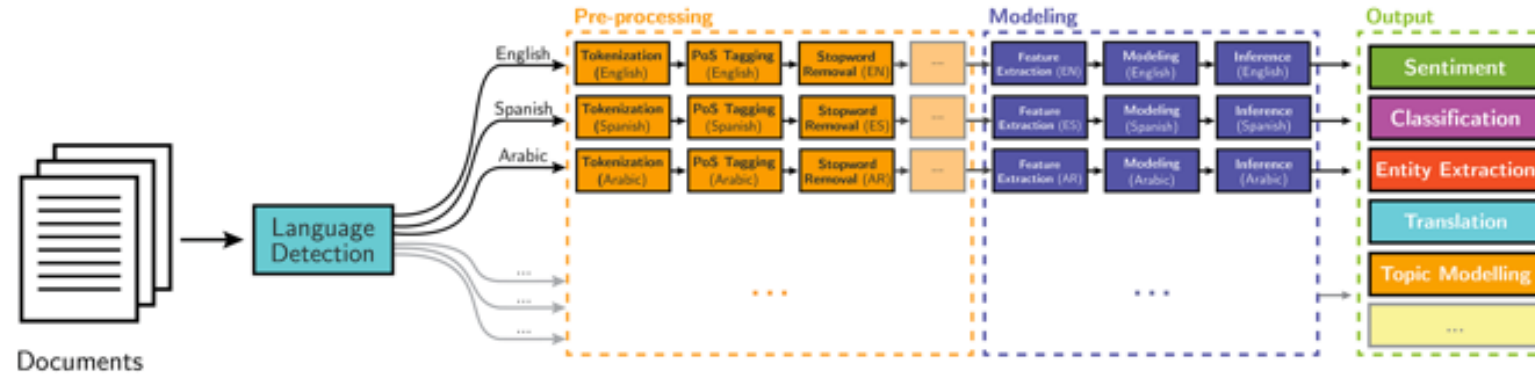
- **Lexical similarity**
 - **Syntax, structure, and content of the documents**
- **Semantic similarity**
 - **Semantics, meaning, and context of the documents**

Cosine Similarity

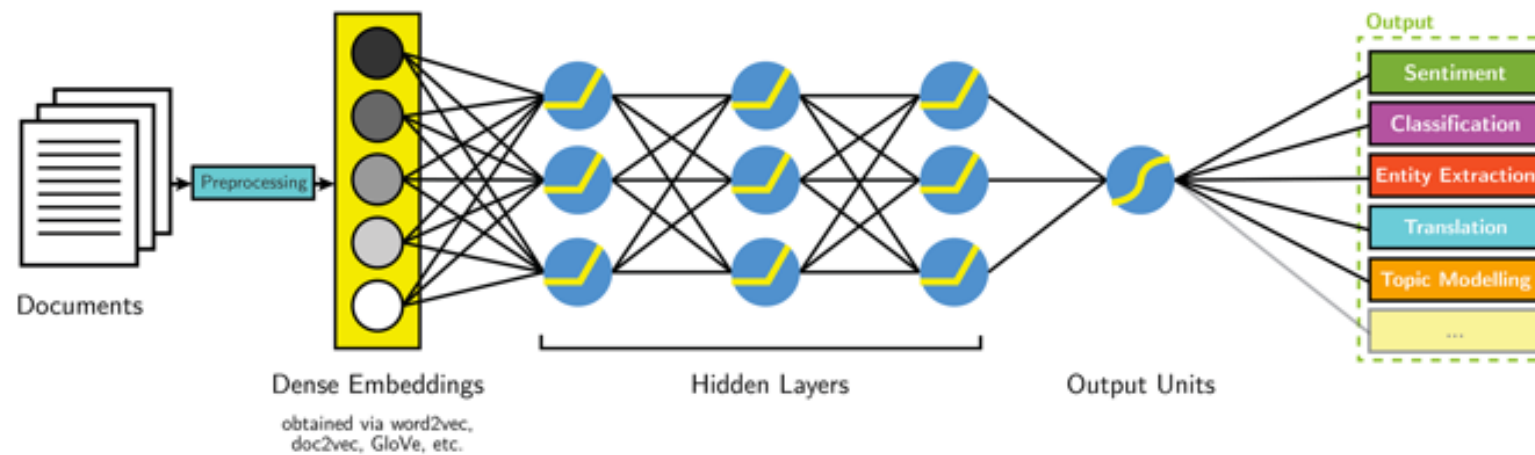


NLP

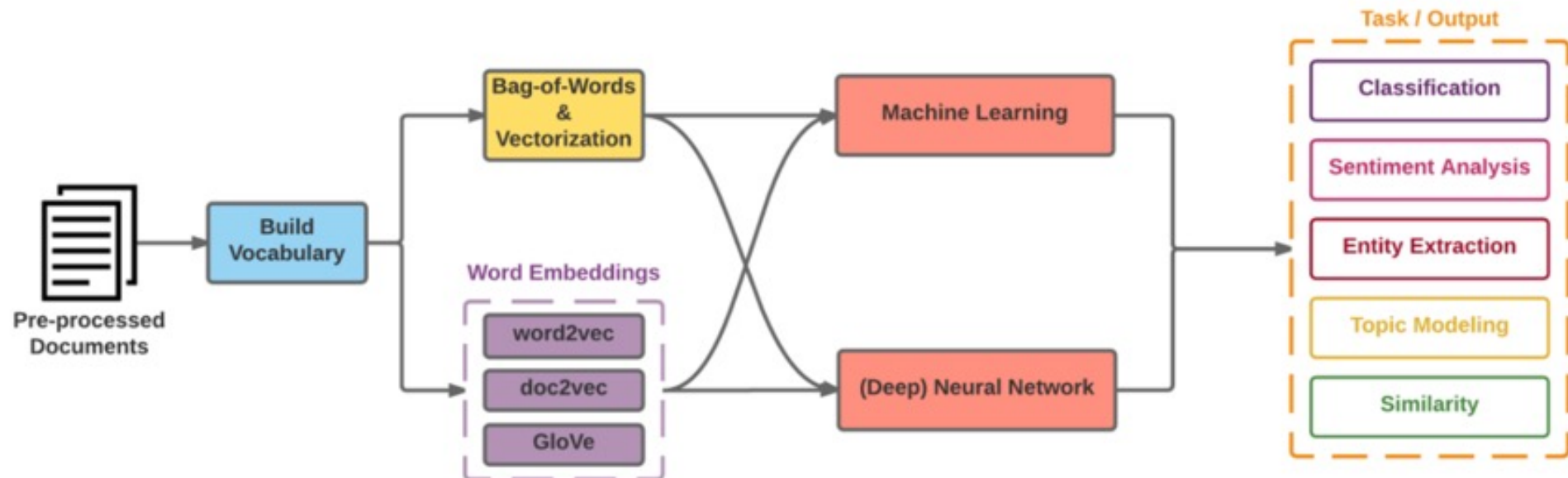
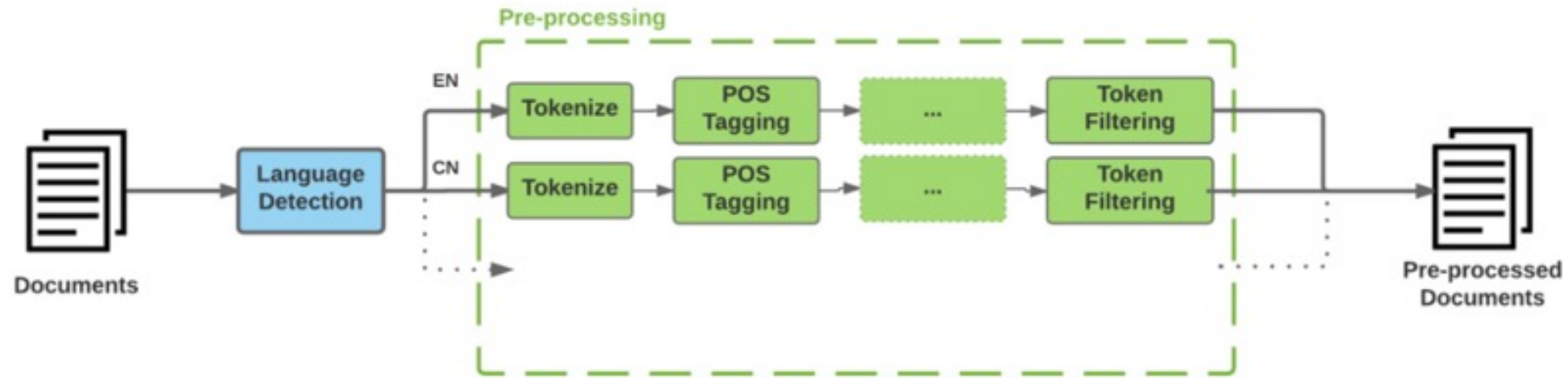
Classical NLP



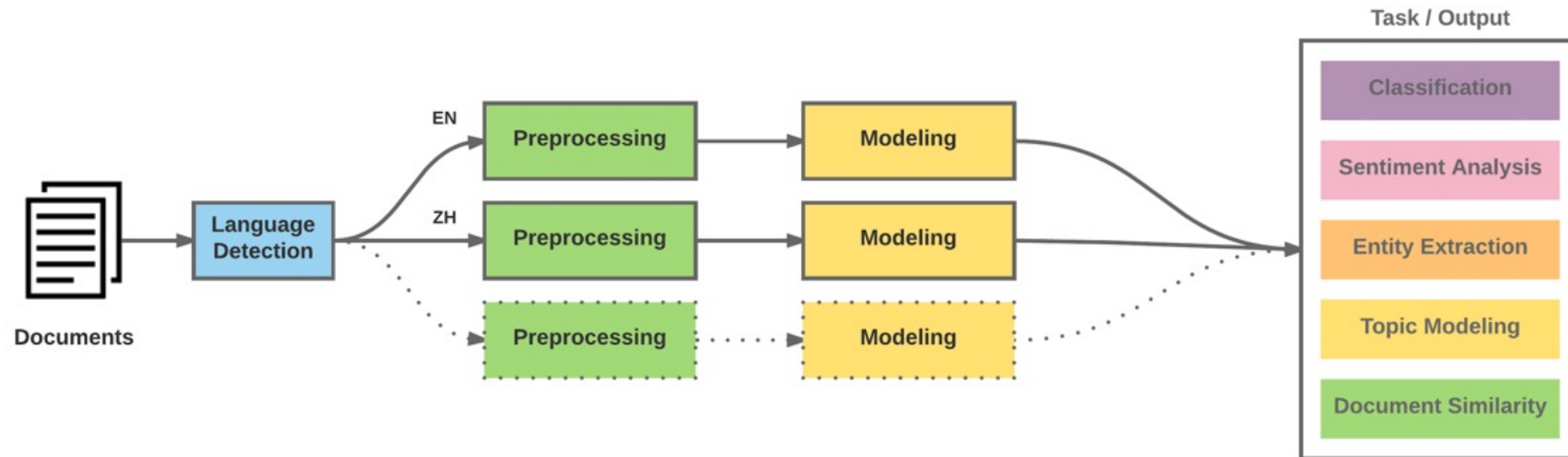
Deep Learning-based NLP



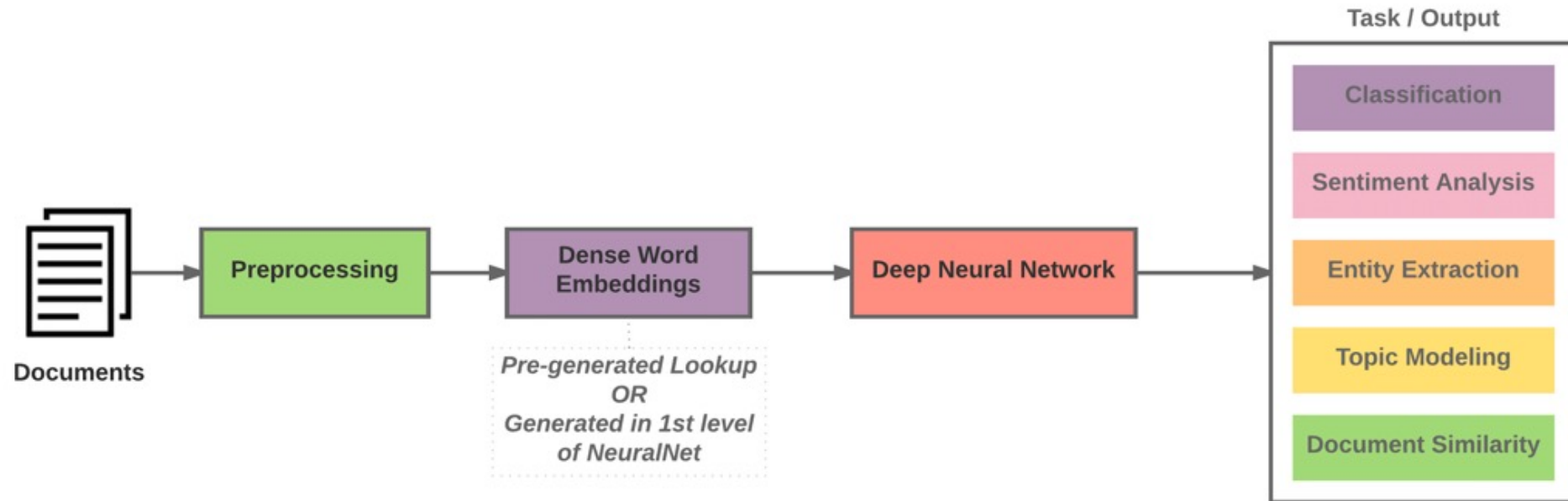
Modern NLP Pipeline



Modern NLP Pipeline



Deep Learning NLP



Natural Language Processing (NLP) and Text Mining

Raw text

Sentence Segmentation

Tokenization

Part-of-Speech (POS)

Stop word removal

Stemming / **Lemmatization**

Dependency Parser

String Metrics & Matching

word's stem

am → am

having → hav

word's lemma

am → be

having → have

Data Mining Tasks & Methods

Data Mining Tasks & Methods	Data Mining Algorithms	Learning Type
Prediction		
Classification	Decision Trees, Neural Networks, Support Vector Machines, kNN, Naïve Bayes, GA	Supervised
Regression	Linear/Nonlinear Regression, ANN, Regression Trees, SVM, kNN, GA	Supervised
Time series	Autoregressive Methods, Averaging Methods, Exponential Smoothing, ARIMA	Supervised
Association		
Market-basket	Apriori, OneR, ZeroR, Eclat, GA	Unsupervised
Link analysis	Expectation Maximization, Apriori Algorithm, Graph-Based Matching	Unsupervised
Sequence analysis	Apriori Algorithm, FP-Growth, Graph-Based Matching	Unsupervised
Segmentation		
Clustering	k-means, Expectation Maximization (EM)	Unsupervised
Outlier analysis	k-means, Expectation Maximization (EM)	Unsupervised

Example of Cluster Analysis

Point	P	P(x,y)
p01	a	(3, 4)
p02	b	(3, 6)
p03	c	(3, 8)
p04	d	(4, 5)
p05	e	(4, 7)
p06	f	(5, 1)
p07	g	(5, 5)
p08	h	(7, 3)
p09	i	(7, 5)
p10	j	(8, 5)

K-Means Clustering

Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	1.95	3.78	Cluster1
p02	b	(3, 6)	0.69	4.51	Cluster1
p03	c	(3, 8)	2.27	5.86	Cluster1
p04	d	(4, 5)	0.89	3.13	Cluster1
p05	e	(4, 7)	1.22	4.45	Cluster1
p06	f	(5, 1)	5.01	3.05	Cluster2
p07	g	(5, 5)	1.57	2.30	Cluster1
p08	h	(7, 3)	4.37	0.56	Cluster2
p09	i	(7, 5)	3.43	1.52	Cluster2
p10	j	(8, 5)	4.41	1.95	Cluster2

m1 (3.67, 5.83)

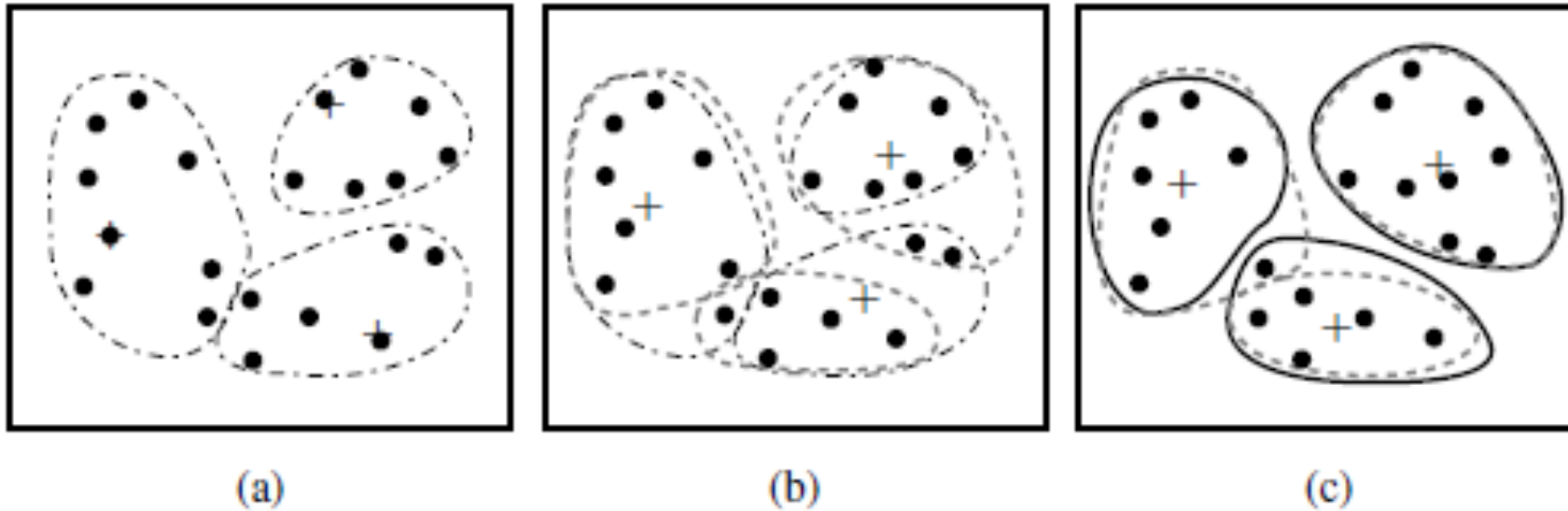
m2 (6.75, 3.50)

Cluster Analysis

Cluster Analysis

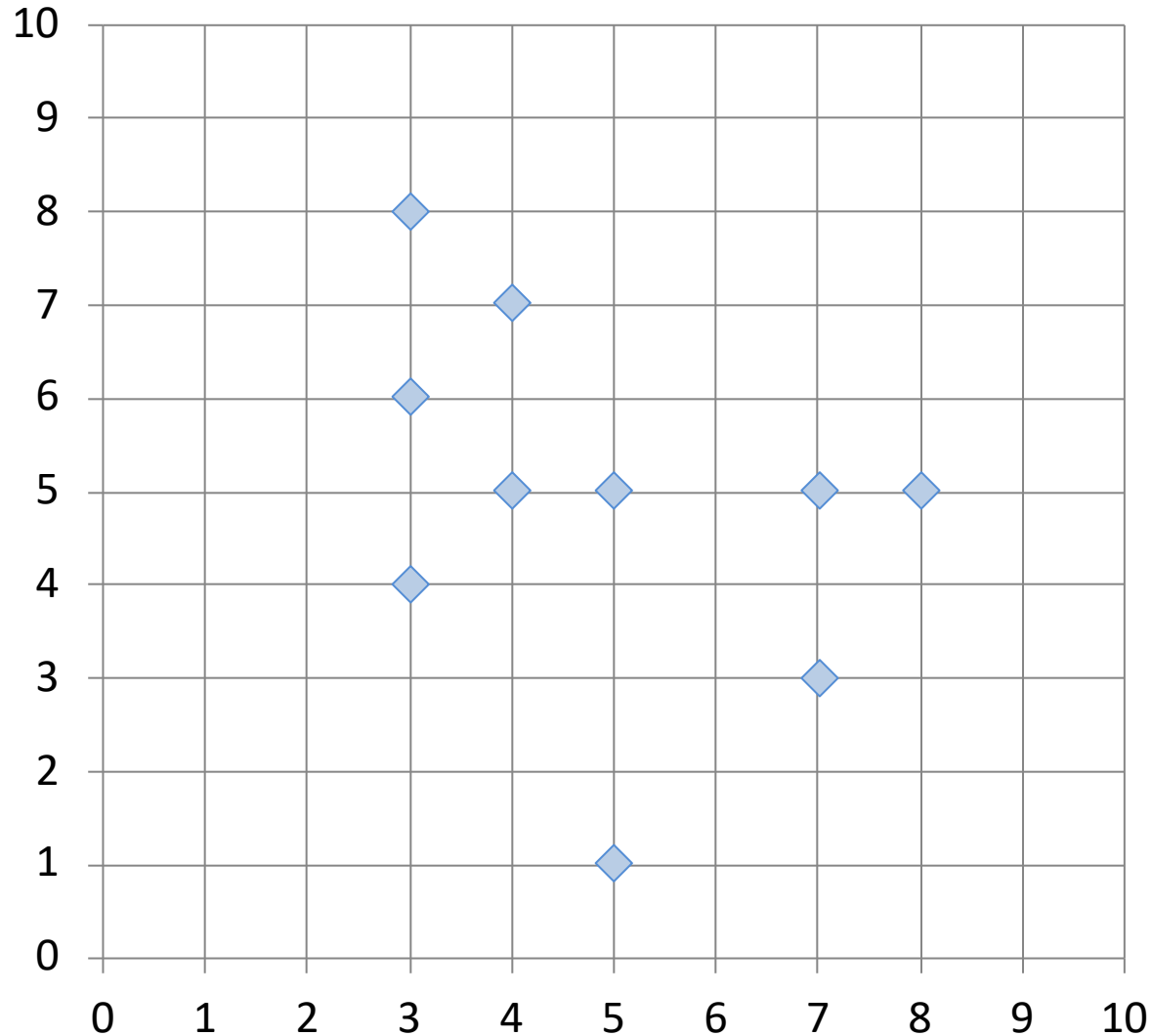
- Used for automatic identification of **natural groupings** of things
- Part of the machine-learning family
- Employ **unsupervised learning**
- Learns the clusters of things from past data, then assigns new instances
- There is not an output variable
- Also known as **segmentation**

Cluster Analysis



Clustering of a set of objects based on the *k-means method*.
(The mean of each cluster is marked by a “+”.)

Example of Cluster Analysis



Point	P	P(x,y)
p01	a	(3, 4)
p02	b	(3, 6)
p03	c	(3, 8)
p04	d	(4, 5)
p05	e	(4, 7)
p06	f	(5, 1)
p07	g	(5, 5)
p08	h	(7, 3)
p09	i	(7, 5)
p10	j	(8, 5)

Cluster Analysis for Data Mining

- **How many clusters?**
 - There is not a “truly optimal” way to calculate it
 - Heuristics are often used
 1. Look at the sparseness of clusters
 2. **Number of clusters** = $(n/2)^{1/2}$ (n: no of data points)
 3. Use Akaike information criterion (AIC)
 4. Use Bayesian information criterion (BIC)
- **Most cluster analysis methods involve the use of a **distance measure** to calculate the closeness between pairs of items**
 - **Euclidian** versus **Manhattan** (rectilinear) **distance**

***k*-Means Clustering Algorithm**

- ***k*** : pre-determined number of clusters
- Algorithm (**Step 0**: determine value of ***k***)

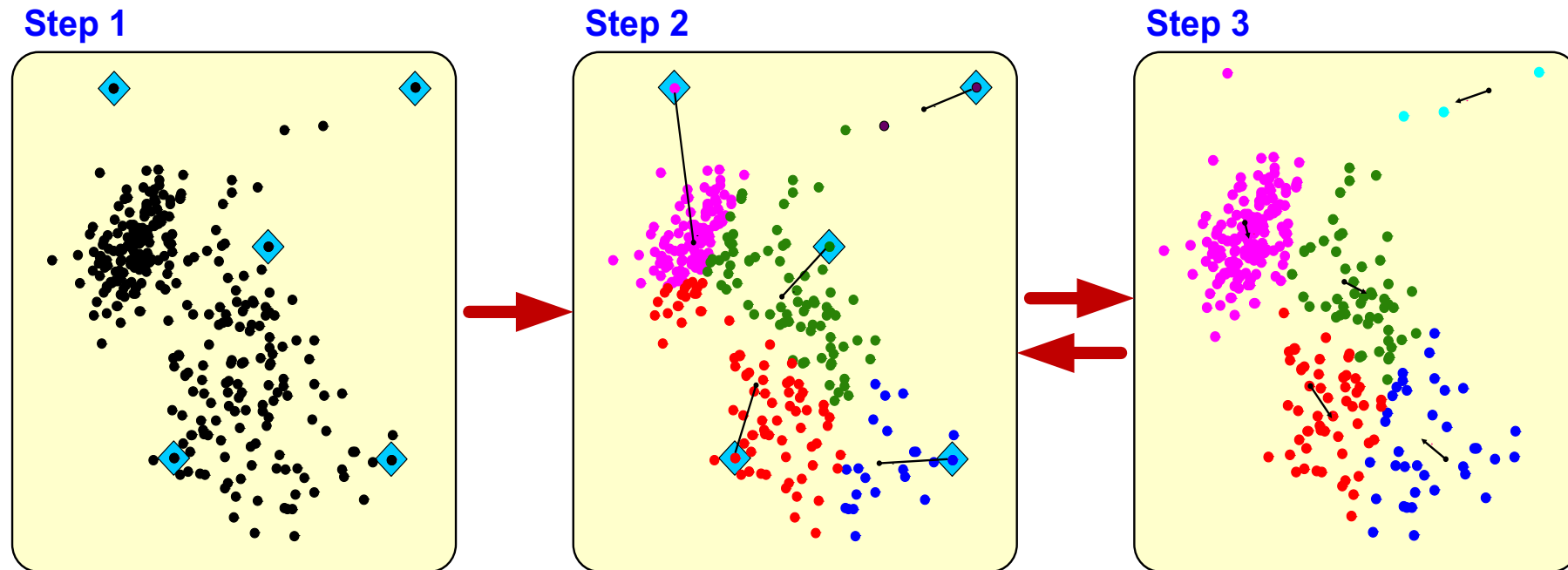
Step 1: Randomly generate ***k*** random points as initial cluster centers

Step 2: Assign each point to the nearest cluster center

Step 3: Re-compute the new cluster centers

Repetition step: Repeat steps 2 and 3 until some convergence criterion is met (usually that the assignment of points to clusters becomes stable)

Cluster Analysis for Data Mining - *k*-Means Clustering Algorithm



Similarity

Distance

Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects

- Some popular ones include: **Minkowski distance**:

$$d(i, j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \dots + |x_{ip} - x_{jp}|^q)}$$

where $i = (x_{i1}, x_{i2}, \dots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \dots, x_{jp})$ are two p -dimensional data objects, and q is a positive integer

- If $q = 1$, d is **Manhattan distance**

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

Similarity and Dissimilarity Between Objects (Cont.)

- If $q = 2$, d is **Euclidean distance**:

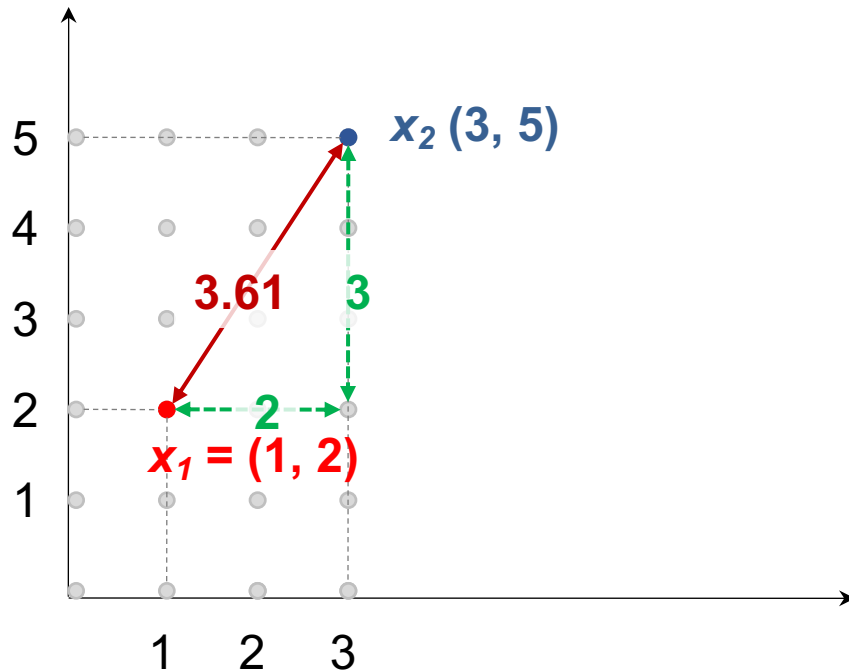
$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{ip} - x_{jp}|^2)}$$

- **Properties**

- $d(i,j) \geq 0$
 - $d(i,i) = 0$
 - $d(i,j) = d(j,i)$
 - $d(i,j) \leq d(i,k) + d(k,j)$
- Also, one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures

Euclidean distance vs Manhattan distance

- Distance of two point $x_1 = (1, 2)$ and $x_2 (3, 5)$



Euclidean distance:

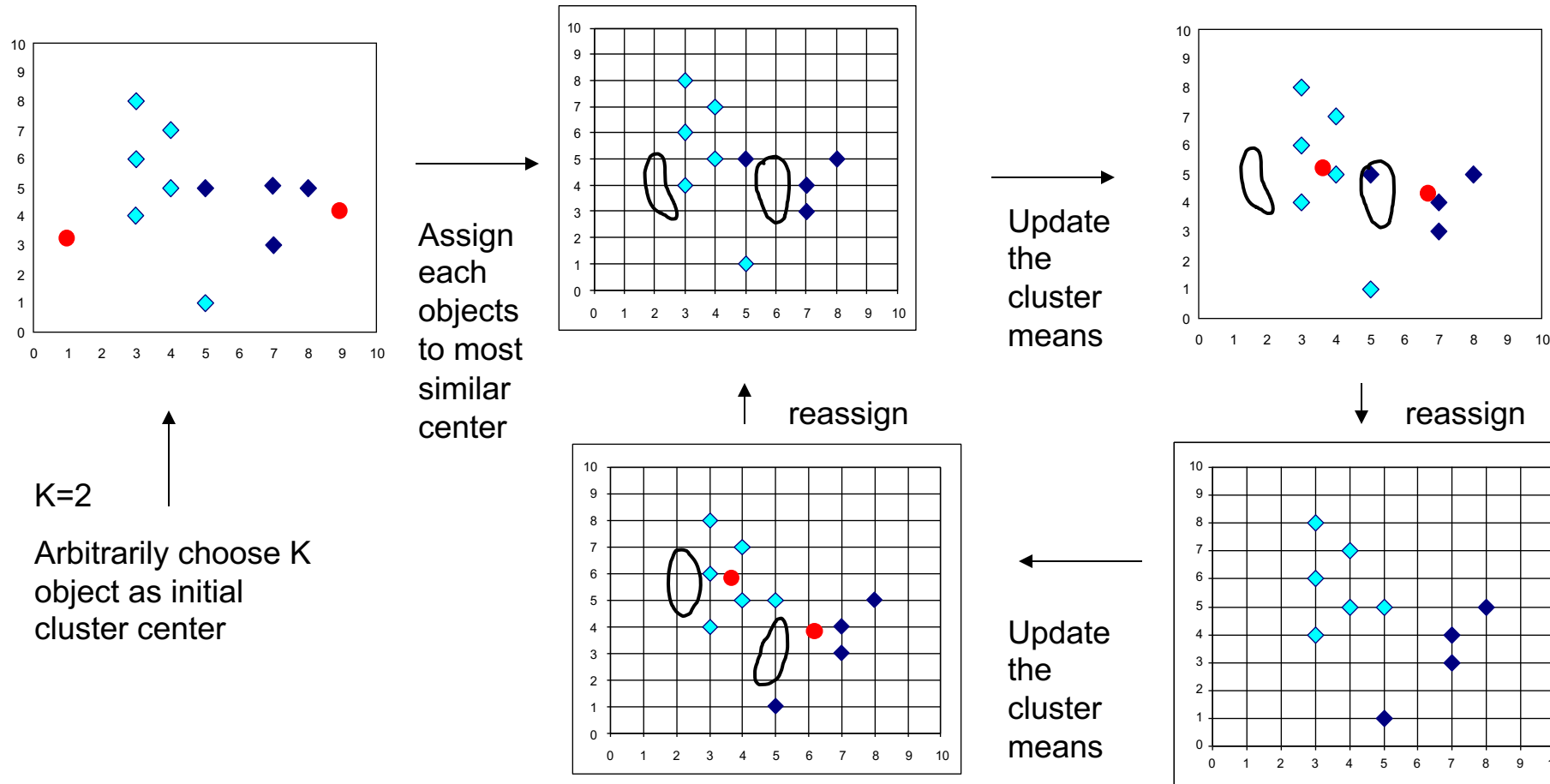
$$\begin{aligned} &= ((3-1)^2 + (5-2)^2)^{1/2} \\ &= (2^2 + 3^2)^{1/2} \\ &= (4 + 9)^{1/2} \\ &= (13)^{1/2} \\ &= 3.61 \end{aligned}$$

Manhattan distance:

$$\begin{aligned} &= (3-1) + (5-2) \\ &= 2 + 3 \\ &= 5 \end{aligned}$$

The *K-Means* Clustering Method

- **Example**



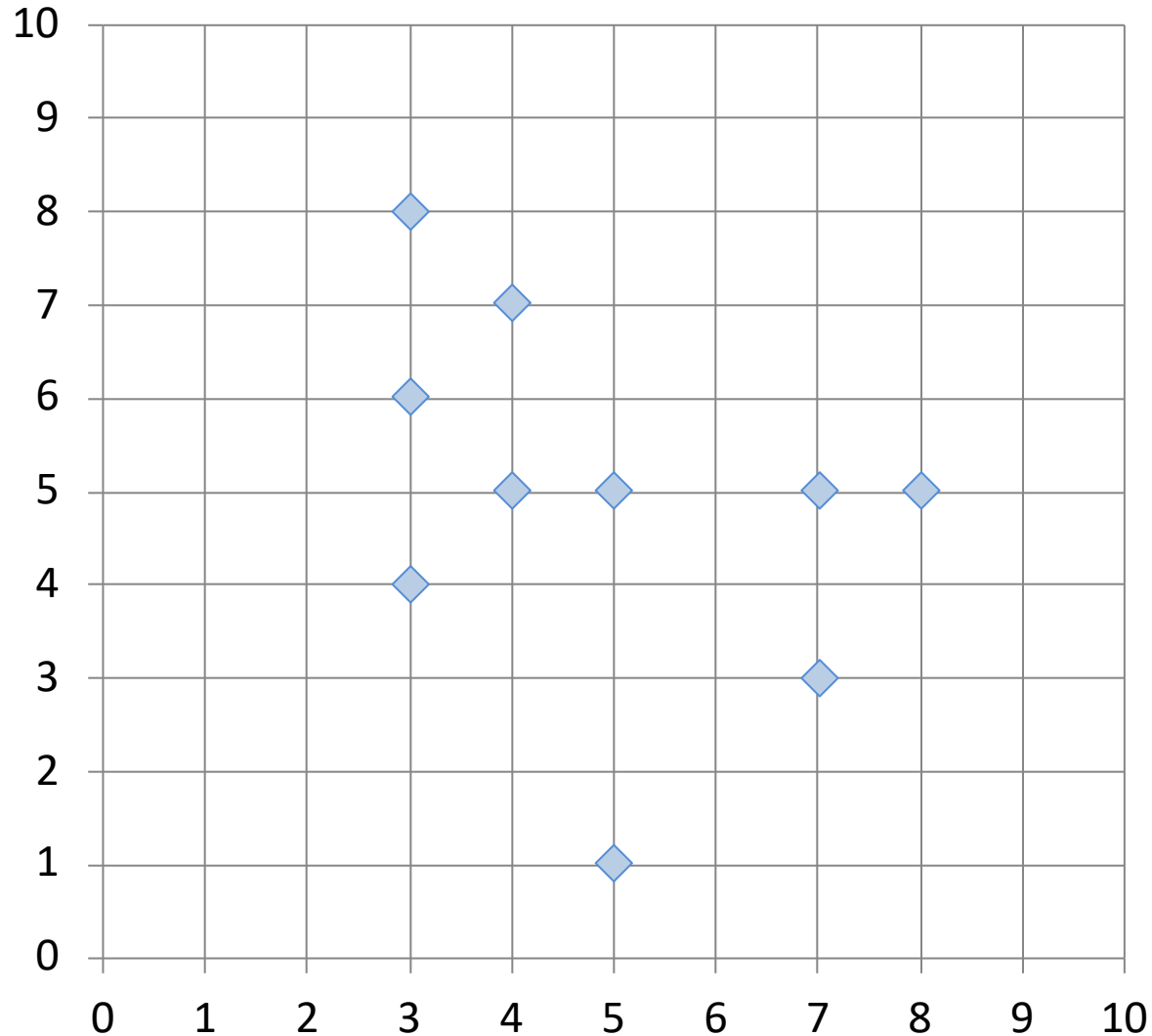
K-Means Clustering

Example of Cluster Analysis

Point	P	P(x,y)
p01	a	(3, 4)
p02	b	(3, 6)
p03	c	(3, 8)
p04	d	(4, 5)
p05	e	(4, 7)
p06	f	(5, 1)
p07	g	(5, 5)
p08	h	(7, 3)
p09	i	(7, 5)
p10	j	(8, 5)

K-Means Clustering

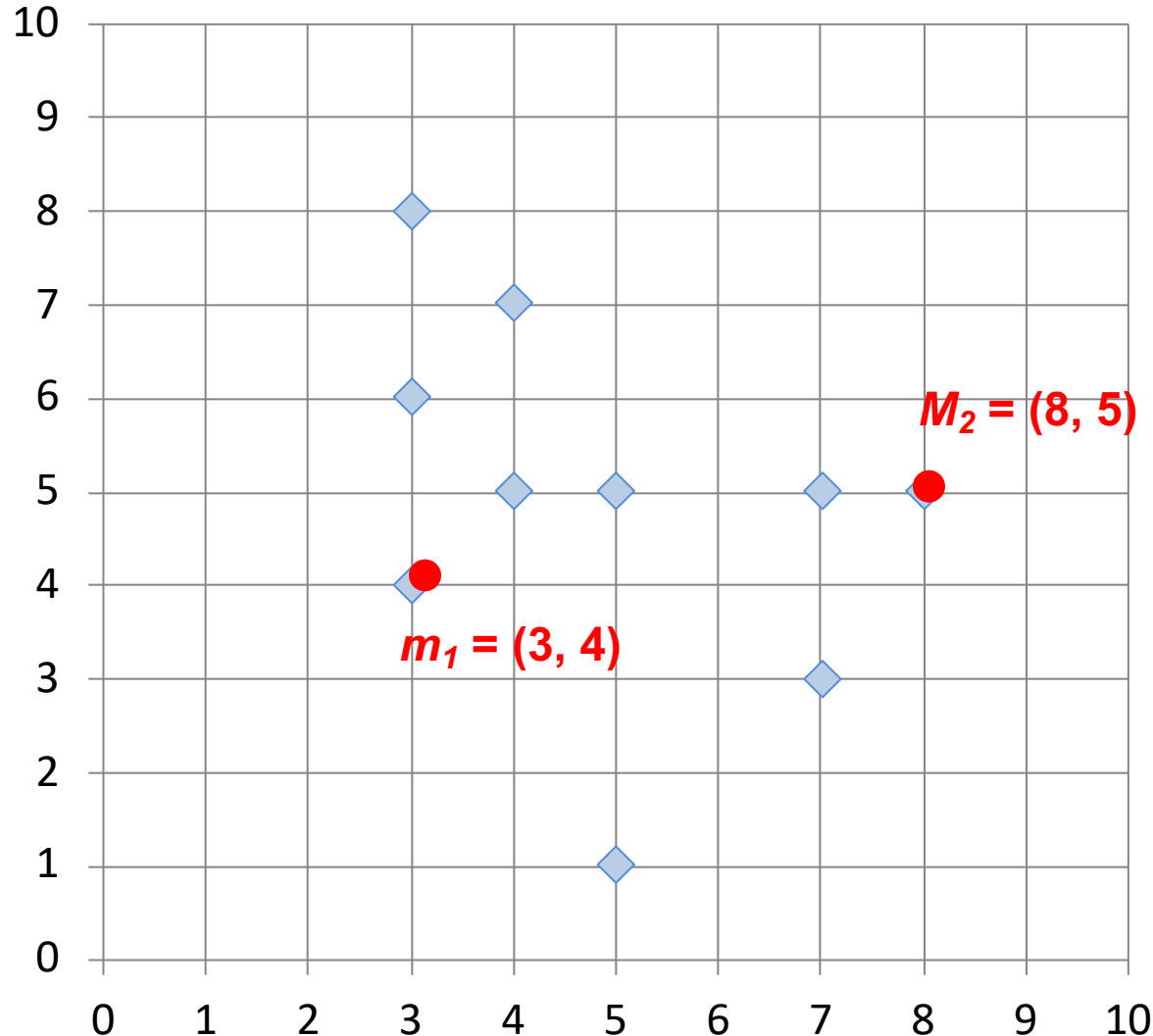
Step by Step



Point	P	P(x,y)
p01	a	(3, 4)
p02	b	(3, 6)
p03	c	(3, 8)
p04	d	(4, 5)
p05	e	(4, 7)
p06	f	(5, 1)
p07	g	(5, 5)
p08	h	(7, 3)
p09	i	(7, 5)
p10	j	(8, 5)

K-Means Clustering

Step 1: K=2, Arbitrarily choose K object as initial cluster center

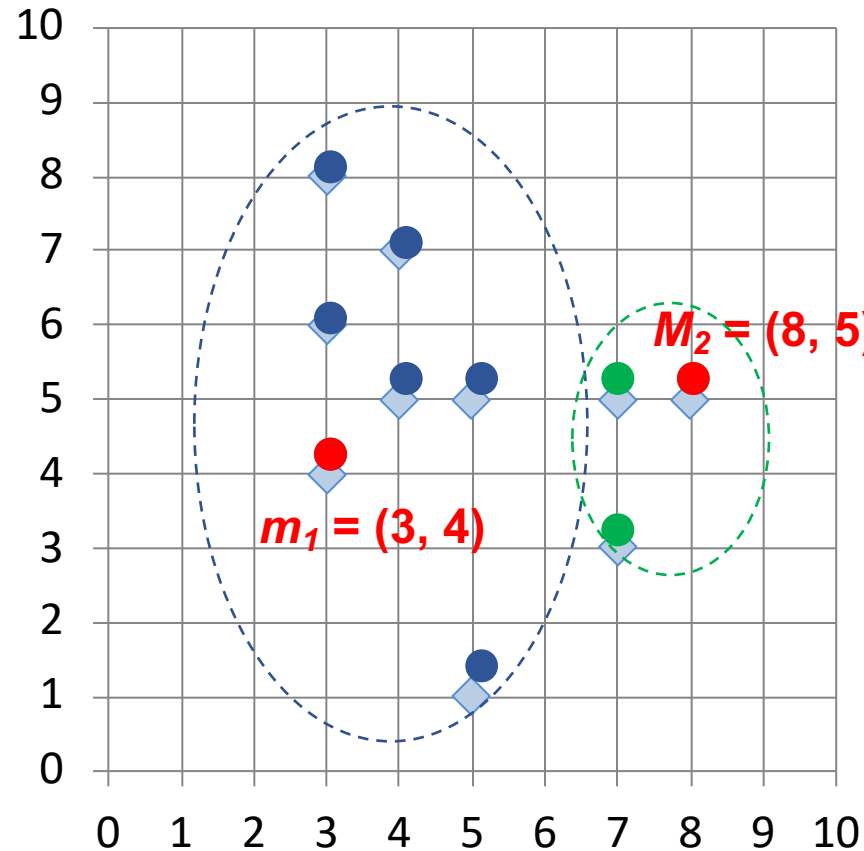


Point	P	P(x,y)
p01	a	(3, 4)
p02	b	(3, 6)
p03	c	(3, 8)
p04	d	(4, 5)
p05	e	(4, 7)
p06	f	(5, 1)
p07	g	(5, 5)
p08	h	(7, 3)
p09	i	(7, 5)
p10	j	(8, 5)

Initial m_1 (3, 4)
Initial m_2 (8, 5)

Step 2: Compute seed points as the centroids of the clusters of the current partition

Step 3: Assign each objects to most similar center



Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	0.00	5.10	Cluster1
p02	b	(3, 6)	2.00	5.10	Cluster1
p03	c	(3, 8)	4.00	5.83	Cluster1
p04	d	(4, 5)	1.41	4.00	Cluster1
p05	e	(4, 7)	3.16	4.47	Cluster1
p06	f	(5, 1)	3.61	5.00	Cluster1
p07	g	(5, 5)	2.24	3.00	Cluster1
p08	h	(7, 3)	4.12	2.24	Cluster2
p09	i	(7, 5)	4.12	1.00	Cluster2
p10	j	(8, 5)	5.10	0.00	Cluster2

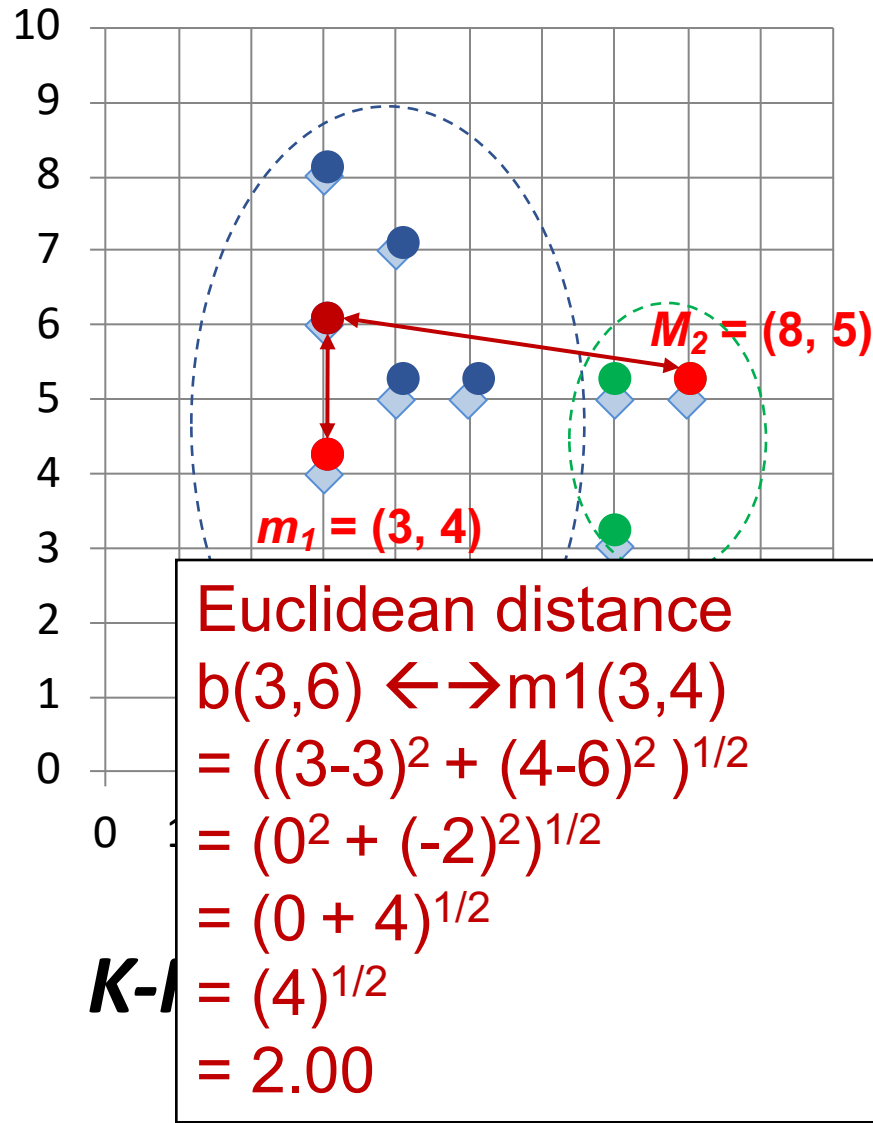
***K-Means* Clustering**

Initial m_1 (3, 4)

Initial m_2 (8, 5)

Step 2: Compute seed points as the centroids of the clusters of the current partition

Step 3: Assign each objects to most similar center



Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	0.00	5.10	Cluster1
p02	b	(3, 6)	2.00	5.10	Cluster1
p03	c	(3, 8)	4.00	5.83	Cluster1
p04	d	(4, 5)	1.41	4.00	Cluster1

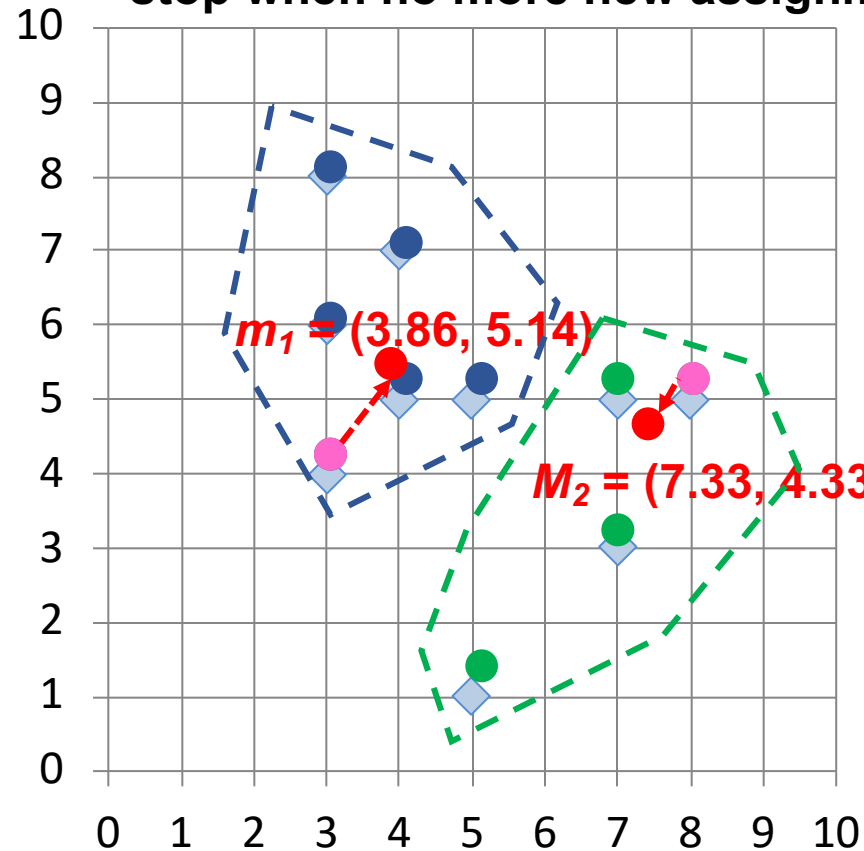
p05					ster1
p06					ster1
p07					ster1
p08					ster2
p09					ster2
p10					ster2

Euclidean distance
 $b(3,6) \leftrightarrow m_2(8,5)$
 $= ((8-3)^2 + (5-6)^2)^{1/2}$
 $= (5^2 + (-1)^2)^{1/2}$
 $= (25 + 1)^{1/2}$
 $= (26)^{1/2}$
 $= 5.10$

Initial m1 (3, 4)

Initial m2 (8, 5)

**Step 4: Update the cluster means,
Repeat Step 2, 3,
stop when no more new assignment**



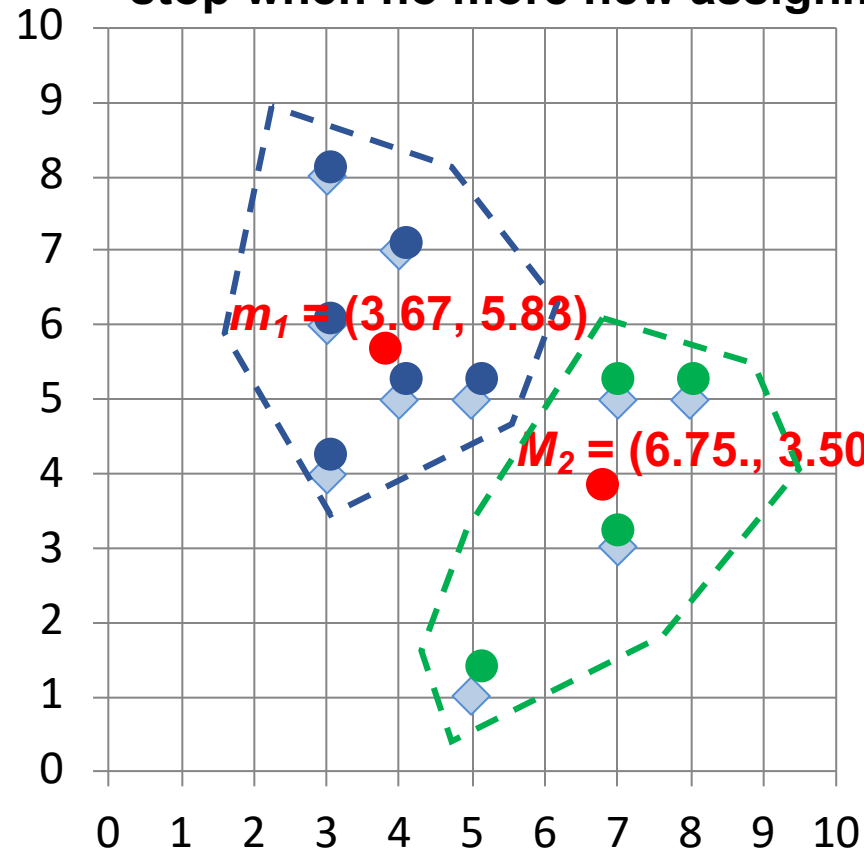
Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	1.43	4.34	Cluster1
p02	b	(3, 6)	1.22	4.64	Cluster1
p03	c	(3, 8)	2.99	5.68	Cluster1
p04	d	(4, 5)	0.20	3.40	Cluster1
p05	e	(4, 7)	1.87	4.27	Cluster1
p06	f	(5, 1)	4.29	4.06	Cluster2
p07	g	(5, 5)	1.15	2.42	Cluster1
p08	h	(7, 3)	3.80	1.37	Cluster2
p09	i	(7, 5)	3.14	0.75	Cluster2
p10	j	(8, 5)	4.14	0.95	Cluster2

m1 (3.86, 5.14)

m2 (7.33, 4.33)

***K-Means* Clustering**

**Step 4: Update the cluster means,
Repeat Step 2, 3,
stop when no more new assignment**

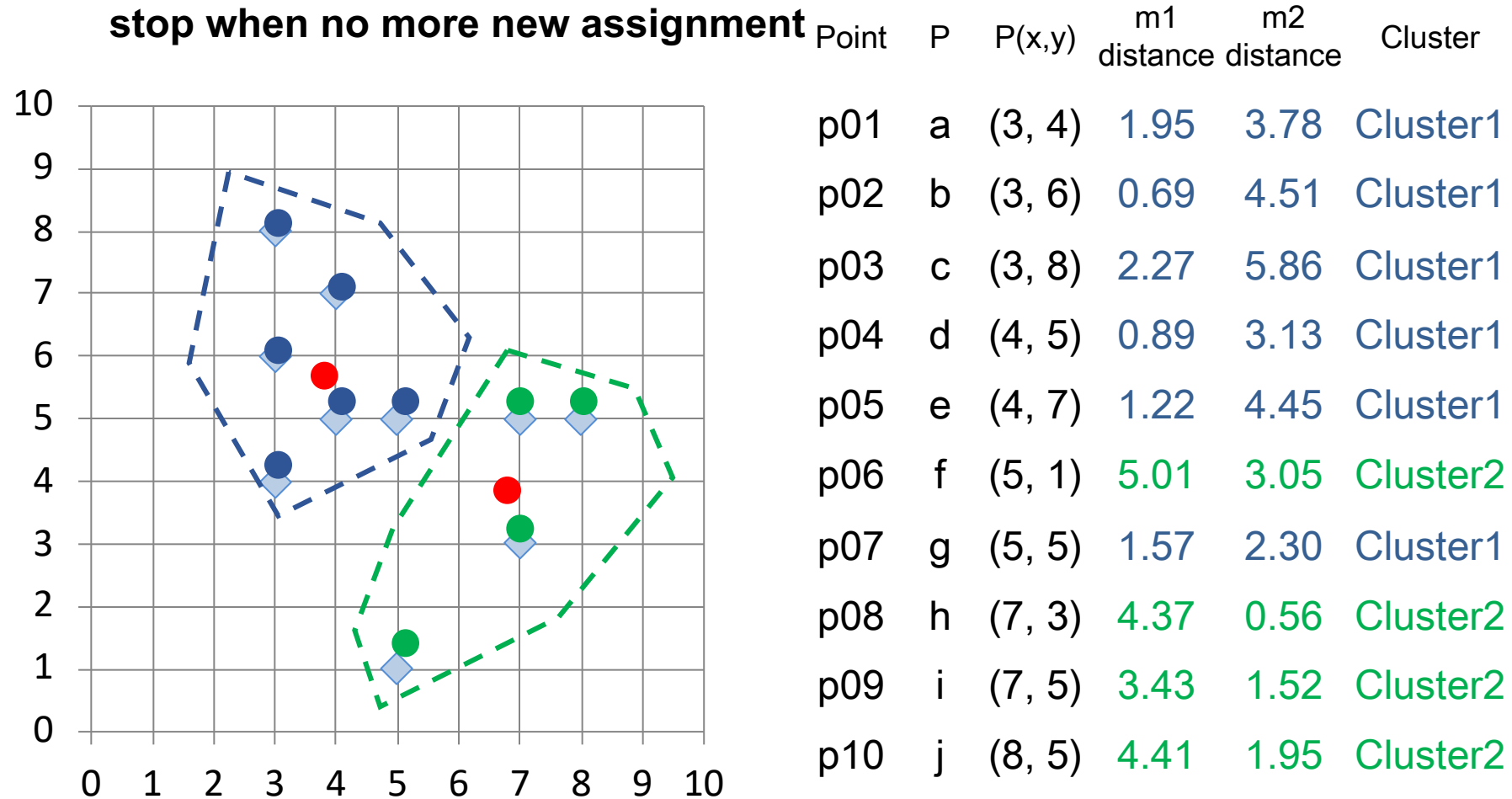


***K-Means* Clustering**

Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	1.95	3.78	Cluster1
p02	b	(3, 6)	0.69	4.51	Cluster1
p03	c	(3, 8)	2.27	5.86	Cluster1
p04	d	(4, 5)	0.89	3.13	Cluster1
p05	e	(4, 7)	1.22	4.45	Cluster1
p06	f	(5, 1)	5.01	3.05	Cluster2
p07	g	(5, 5)	1.57	2.30	Cluster1
p08	h	(7, 3)	4.37	0.56	Cluster2
p09	i	(7, 5)	3.43	1.52	Cluster2
p10	j	(8, 5)	4.41	1.95	Cluster2

m_1 (3.67, 5.83)

m_2 (6.75, 3.50)



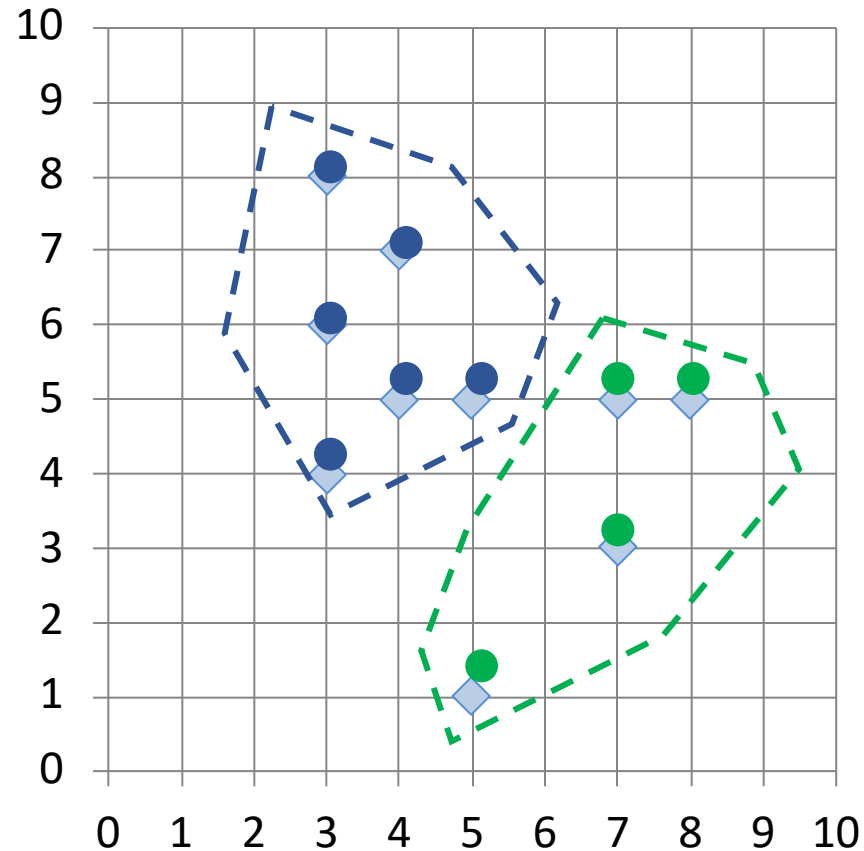
K-Means Clustering

m1 (3.67, 5.83)

m2 (6.75, 3.50)

K-Means Clustering ($K=2$, two clusters)

stop when no more new assignment



Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	1.95	3.78	Cluster1
p02	b	(3, 6)	0.69	4.51	Cluster1
p03	c	(3, 8)	2.27	5.86	Cluster1
p04	d	(4, 5)	0.89	3.13	Cluster1
p05	e	(4, 7)	1.22	4.45	Cluster1
p06	f	(5, 1)	5.01	3.05	Cluster2
p07	g	(5, 5)	1.57	2.30	Cluster1
p08	h	(7, 3)	4.37	0.56	Cluster2
p09	i	(7, 5)	3.43	1.52	Cluster2
p10	j	(8, 5)	4.41	1.95	Cluster2

K-Means Clustering

m1 (3.67, 5.83)

m2 (6.75, 3.50)

K-Means Clustering

Point	P	P(x,y)	m1 distance	m2 distance	Cluster
p01	a	(3, 4)	1.95	3.78	Cluster1
p02	b	(3, 6)	0.69	4.51	Cluster1
p03	c	(3, 8)	2.27	5.86	Cluster1
p04	d	(4, 5)	0.89	3.13	Cluster1
p05	e	(4, 7)	1.22	4.45	Cluster1
p06	f	(5, 1)	5.01	3.05	Cluster2
p07	g	(5, 5)	1.57	2.30	Cluster1
p08	h	(7, 3)	4.37	0.56	Cluster2
p09	i	(7, 5)	3.43	1.52	Cluster2
p10	j	(8, 5)	4.41	1.95	Cluster2

m1 (3.67, 5.83)

m2 (6.75, 3.50)

gensim



gensim

topic modelling for humans

[Fork me on GitHub](#)

[Download](#)
latest version from the Python Package Index

[Direct install with:
easy_install -U gensim](#)

[Home](#) [Tutorials](#) [Install](#) [Support](#) [API](#) [About](#)

```
>>> from gensim import corpora, models, similarities
>>>
>>> # Load corpus iterator from a Matrix Market file on disk.
>>> corpus = corpora.MmCorpus('/path/to/corpus.mm')
>>>
>>> # Initialize Latent Semantic Indexing with 200 dimensions.
>>> lsi = models.LsiModel(corpus, num_topics=200)
>>>
>>> # Convert another corpus to the latent space and index it.
>>> index = similarities.MatrixSimilarity(lsi[another_corpus])
>>>
>>> # Compute similarity of a query vs. indexed documents
>>> sims = index[query]
```

Gensim is a FREE Python library

- ✓ Scalable statistical semantics
- ✓ Analyze plain-text documents for semantic structure
- ✓ Retrieve semantically similar documents

spaCy

The banner features a blue background with a pattern of white line-art icons representing various concepts in AI, linguistics, and technology, such as neural networks, gears, and document icons. The spaCy logo is in the top left, and navigation links are in the top right. The main title is centered in large white font, and three feature boxes are at the bottom.

spaCy

HOME USAGE API DEMOS BLOG

Industrial-Strength Natural Language Processing in Python

Fastest in the world

spaCy excels at large-scale information extraction tasks. It's written from the ground up in carefully memory-managed Cython. Independent research has confirmed that spaCy is the fastest in the world. If your application needs to process entire web dumps, spaCy is the library you want to be using.

Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive. I like to think of spaCy as the Ruby on Rails of Natural Language Processing.

Deep learning

spaCy is the best way to prepare text for deep learning. It interoperates seamlessly with [TensorFlow](#), [Keras](#), [Scikit-Learn](#), [Gensim](#) and the rest of Python's awesome AI ecosystem. spaCy helps you connect the statistical models trained by these libraries to the rest of your application.

<https://spacy.io/>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

Comment Share ⚙️ A

RAM Disk

Editing

Table of contents

- Build the model
- Train the model
- Evaluate the model
- Create a graph of accuracy and loss over time
- Text Classification: BBC News Articles
- Text Summarization and Topic Modeling
- Text Summarization
 - Text Summarization with Gensim Summarization
- Topic Modeling
 - Topic Modeling with Gensim LSI model
 - Topic Modeling with Gensim LDA model
 - Topic Modeling with Scikit-learn LDA and NMF
 - Topic Modeling Visualization
- Text Similarity and Clustering
 - Text Similarity**
 - Text Clustering
- Data Visualization
- Section

Text Similarity and Clustering

Text Similarity

- Spacy Vectors Similarity: <https://spacy.io/usage/vectors-similarity>

```
[1] 1 !python -m spacy download en_core_web_sm
```

```
[2] 1 !python -m spacy download en_core_web_lg
    2 # Restart Runtime
```

```
[3] 1 import spacy
    2 nlp = spacy.load("en_core_web_lg")
    3 tokens = nlp("apple banana cat dog notaword")
    4 for token in tokens:
    5     print(token.text, token.has_vector, token.vector_norm, token.is_oov)
```

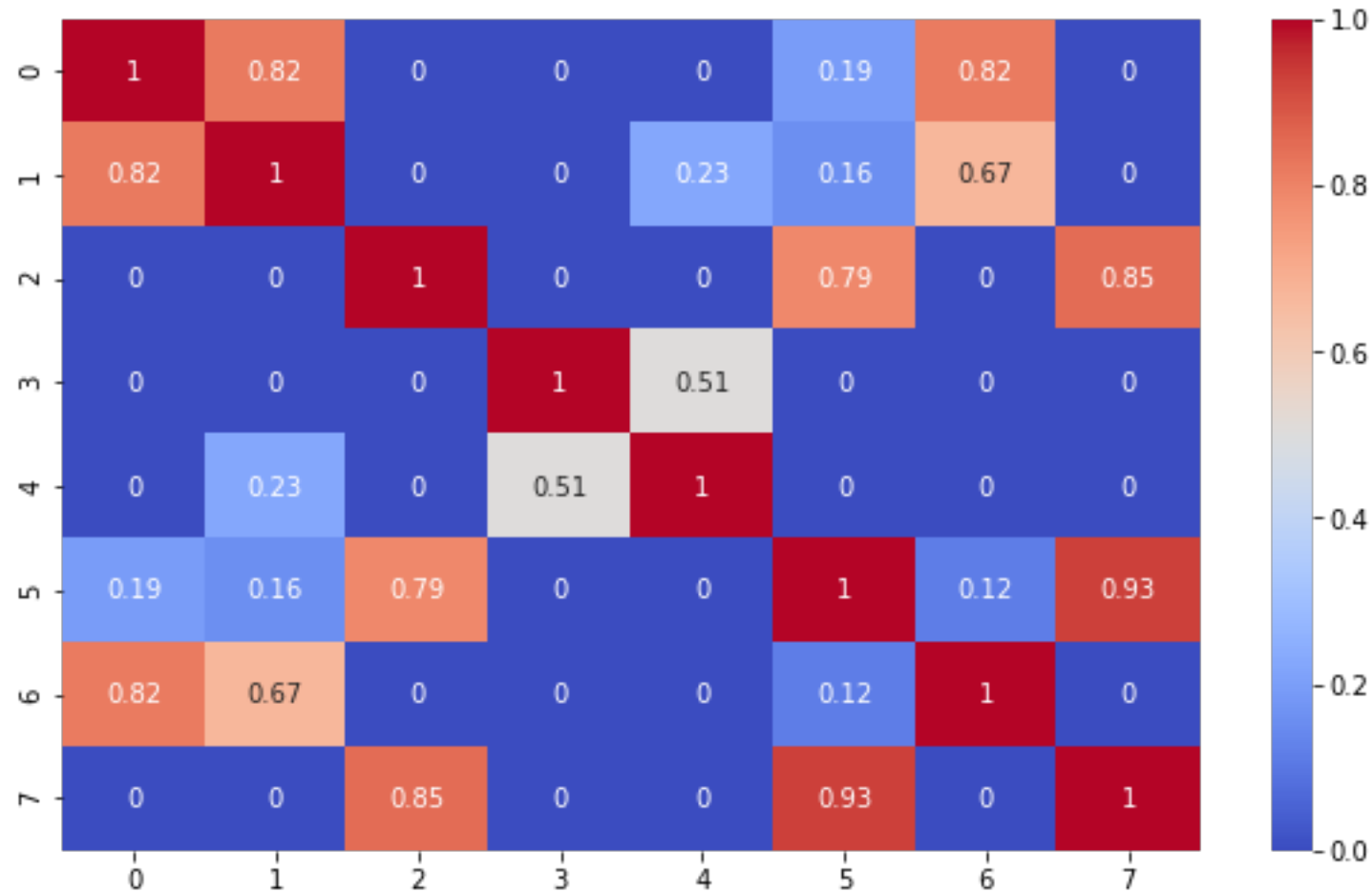
```
apple True 7.1346846 False
banana True 6.700014 False
cat True 6.6808186 False
dog True 7.0336733 False
notaword False 0.0 True
```

```
1 import spacy
2 nlp = spacy.load("en_core_web_lg")
3 doc1 = nlp("I like cat.")
4 doc2 = nlp("I like dog.")
5 doc1.similarity(doc2)
```

<https://tinyurl.com/aintpuppython101>

Python in Google Colab (Python101)

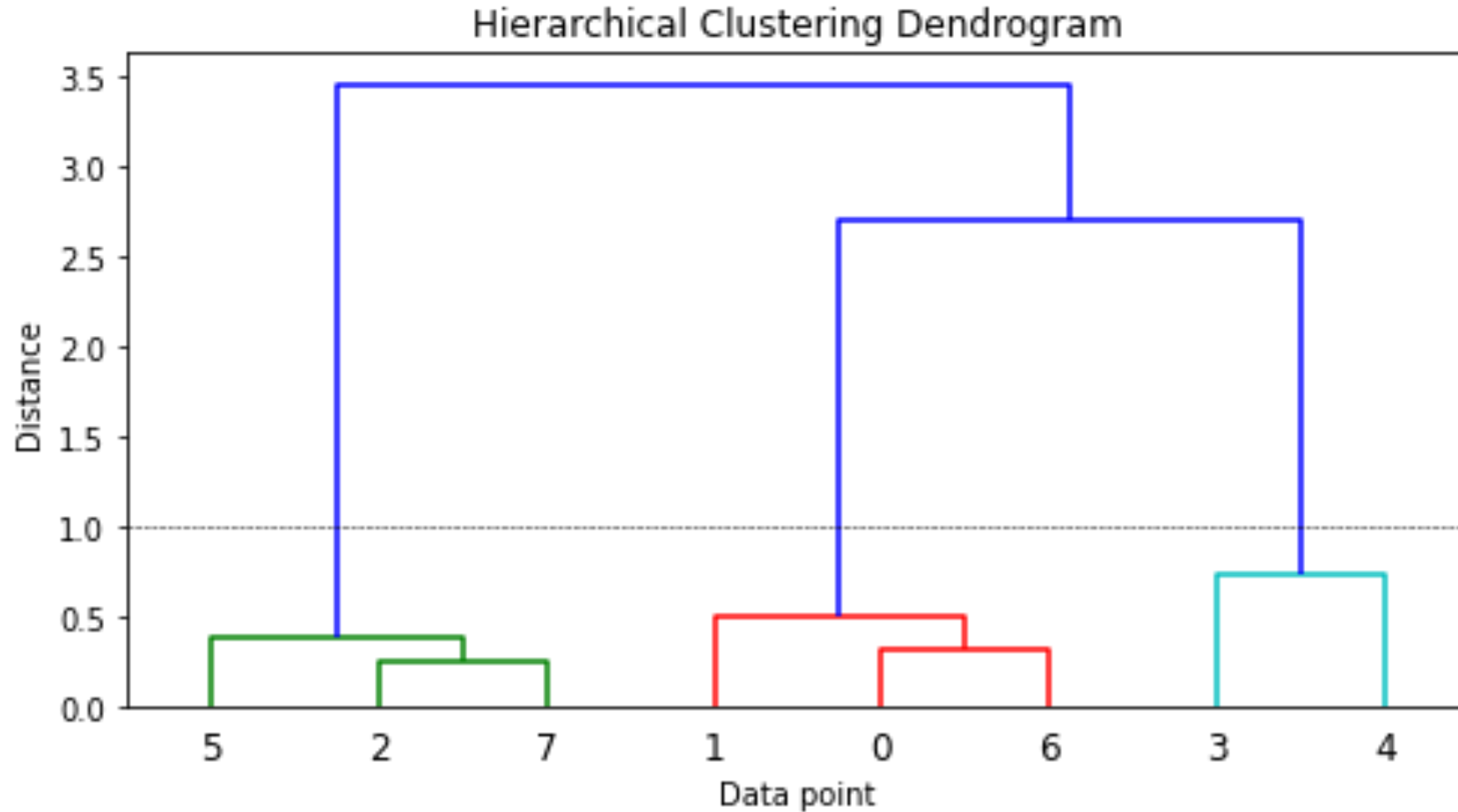
<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



<https://tinyurl.com/aintpuppython101>

Python in Google Colab (Python101)

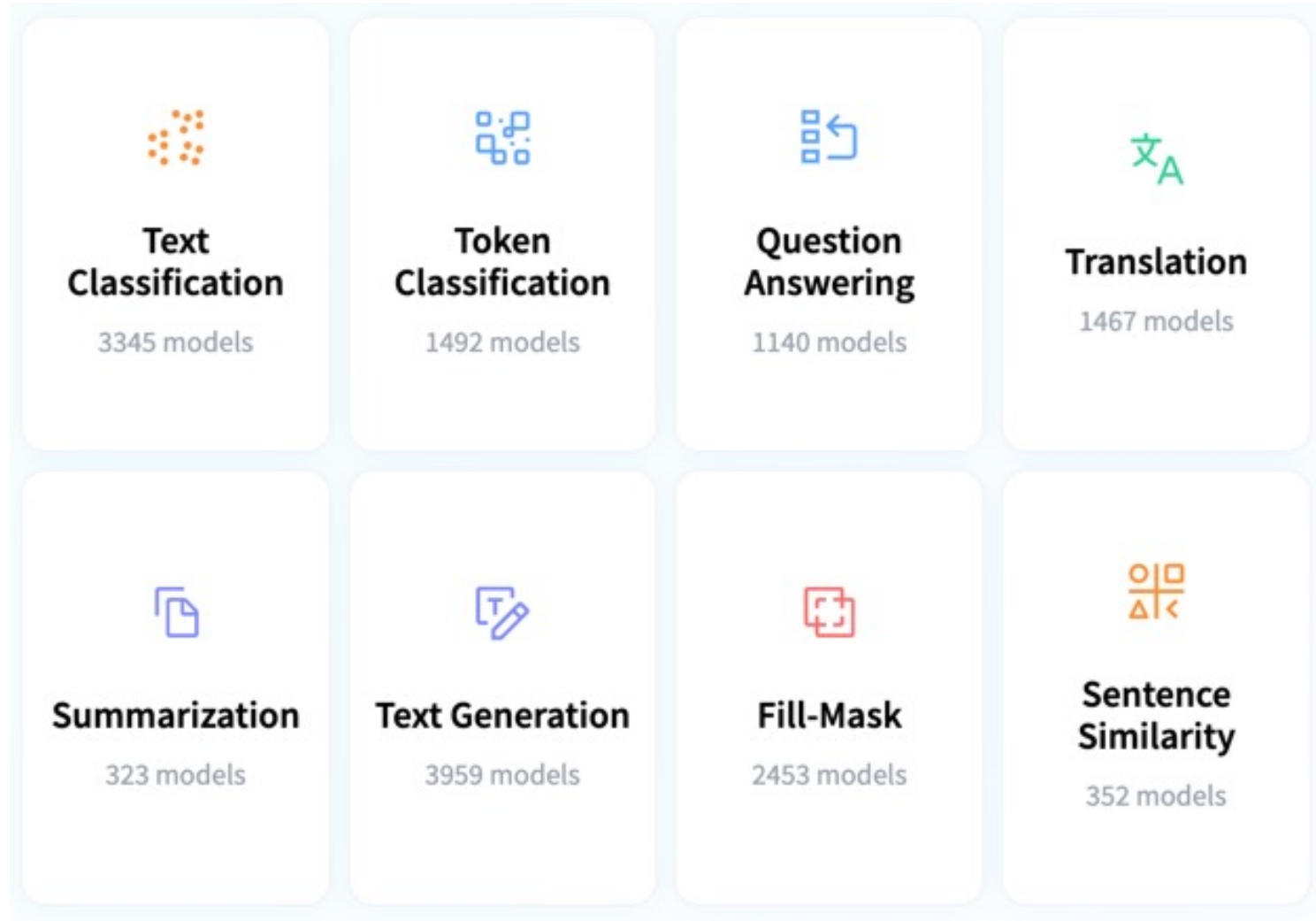
<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



<https://tinyurl.com/aintpuppython101>

Hugging Face Tasks

Natural Language Processing



NLP with Transformers Github

 Why GitHub? ▾ Team Enterprise Explore ▾ Marketplace Pricing ▾

Search / Sign in Sign up

nlp-with-transformers / notebooks Public

Notifications Fork 170 Star 1.1k ▾

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

main ▾ 1 branch 0 tags

Go to file Code ▾

About

Jupyter notebooks for the Natural Language Processing with Transformers book

transformersbook.com/

Readme Apache-2.0 License 1.1k stars 33 watching 170 forks

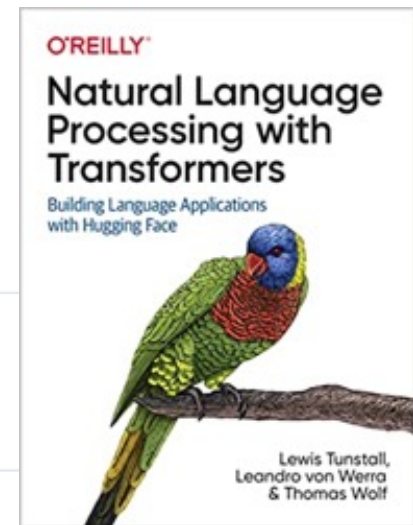
Releases

No releases published

Packages

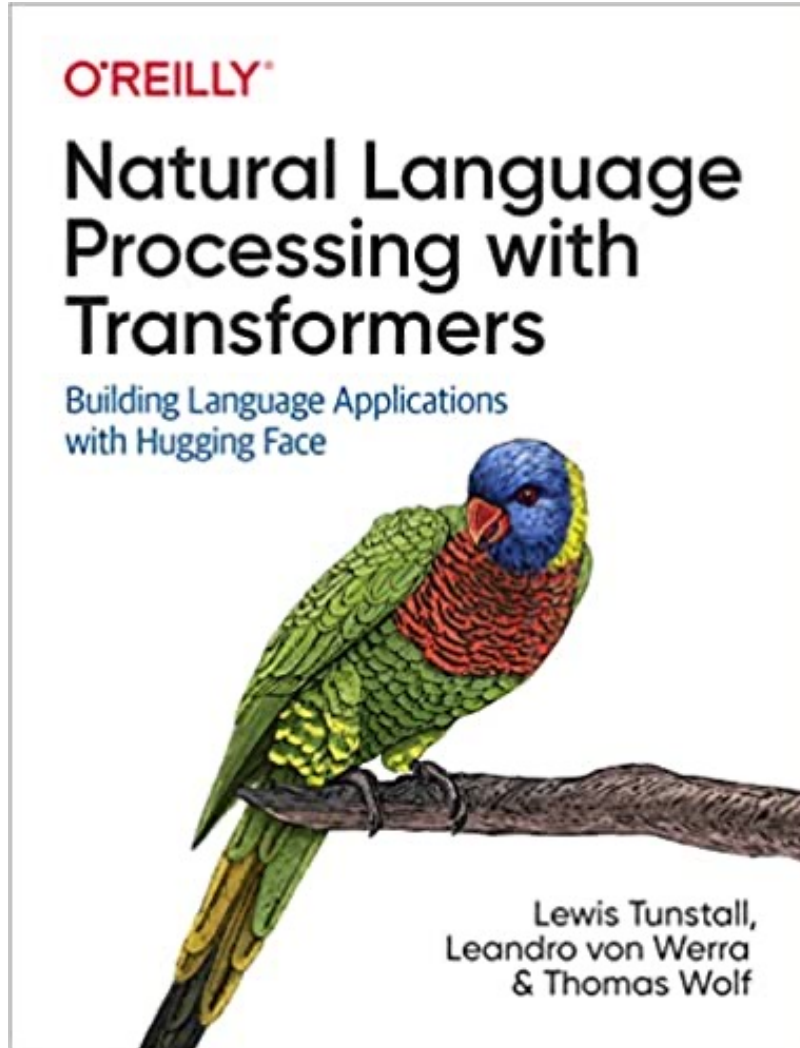
 lewuntun Merge pull request #21 from JingchaoZhang/patch-3 ... ae5b7c1 15 days ago 71 commits

📁 .github/ISSUE_TEMPLATE	Update issue templates	25 days ago
📁 data	Move dataset to data directory	4 months ago
📁 images	Add README	last month
📁 scripts	Update issue templates	25 days ago
📄 .gitignore	Initial commit	4 months ago
📄 01_introduction.ipynb	Remove Colab badges & fastdoc refs	27 days ago
📄 02_classification.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
📄 03_transformer-anatomy.ipynb	[Transformers Anatomy] Remove cells with figure references	22 days ago
📄 04_multilingual-ner.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
📄 05_text-generation.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago



<https://github.com/nlp-with-transformers/notebooks>

NLP with Transformers Github Notebooks



Running on a cloud platform

To run these notebooks on a cloud platform, just click on one of the badges in the table below:

Chapter	Colab	Kaggle	Gradient	Studio Lab
Introduction	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Classification	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Transformer Anatomy	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Multilingual Named Entity Recognition	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Generation	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Summarization	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Question Answering	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Making Transformers Efficient in Production	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Dealing with Few to No Labels	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Training Transformers from Scratch	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Future Directions	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab

Nowadays, the GPUs on Colab tend to be K80s (which have limited memory), so we recommend using [Kaggle](#), [Gradient](#), or [SageMaker Studio Lab](#). These platforms tend to provide more performant GPUs like P100s, all for free!

<https://github.com/nlp-with-transformers/notebooks>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook titled "python101.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A "Table of contents" sidebar on the left lists various topics under three main categories: "Natural Language Processing with Transformers" (including Text Classification, Named Entity Recognition, Question Answering, Summarization, Translation, and Text Generation), "AI in Finance" (including Normative Finance and Financial Theories, Uncertainty and Risk, Expected Utility Theory (EUT), Mean-Variance Portfolio Theory (MVPT), Capital Asset Pricing Model (CAPM), and Arbitrage Pricing Theory (APT)), and "Data Driven Finance" (including Financial Econometrics and Regression, Data Availability, and Normative Theories Revisited). The main notebook area shows three code cells. The first cell clones a GitHub repository and installs requirements. The second cell imports utilities and sets up a chapter. The third cell defines a text variable for a letter to Amazon. Below this, a section titled "Text Classification" shows two more code cells: one for creating a pipeline and another for running the classification and displaying the results as a pandas DataFrame.

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Table of contents

- Natural Language Processing with Transformers
 - Text Classification
 - Named Entity Recognition
 - Question Answering
 - Summarization
 - Translation
 - Text Generation
- AI in Finance
 - Normative Finance and Financial Theories
 - Uncertainty and Risk
 - Expected Utility Theory (EUT)
 - Mean-Variance Portfolio Theory (MVPT)
 - Capital Asset Pricing Model (CAPM)
 - Arbitrage Pricing Theory (APT)
- Data Driven Finance
 - Financial Econometrics and Regression
 - Data Availability
 - Normative Theories Revisited
 - Mean-Variance Portfolio Theory

Natural Language Processing with Transformers

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
[1] 1 !git clone https://github.com/nlp-with-transformers/notebooks.git
    2 %cd notebooks
    3 from install import *
    4 install_requirements()
```

```
[3] 1 from utils import *
    2 setup_chapter()
```

```
[12] 1 text = """Dear Amazon, last week I ordered an Optimus Prime action figure \
      2 from your online store in Germany. Unfortunately, when I opened the package, \
      3 I discovered to my horror that I had been sent an action figure of Megatron \
      4 instead! As a lifelong enemy of the Decepticons, I hope you can understand my \
      5 dilemma. To resolve the issue, I demand an exchange of Megatron for the \
      6 Optimus Prime figure I ordered. Enclosed are copies of my records concerning \
      7 this purchase. I expect to hear from you soon. Sincerely, Bumblebee."""
```

Text Classification

```
[13] 1 from transformers import pipeline
      2 classifier = pipeline("text-classification")
```

```
[14] 1 import pandas as pd
      2 outputs = classifier(text)
      3 pd.DataFrame(outputs)
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings A

RAM Disk Editing

Table of contents

- Text Classification with Transformers
 - The Dataset
 - From Datasets to DataFrames
 - From Text to Tokens
 - Character Tokenization
 - Word Tokenization
 - Subword Tokenization
 - Tokenizing the Whole Dataset
 - Training a Text Classifier
 - Transformers as Feature Extractors
 - Extracting the last hidden states
 - Creating a feature matrix
 - Visualizing the training set
 - Training a simple classifier
 - Fine-Tuning Transformers
 - Loading a pretrained model
 - Defining the performance metrics
 - Training the model
- Sidebar: Fine-Tuning with Keras
- Error analysis
- Saving and sharing the model

Text Classification with Transformers

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
[10] 1 !nvidia-smi
```

```
1 # Uncomment and run this cell if you're on Colab or Kaggle
2 !git clone https://github.com/nlp-with-transformers/notebooks.git
3 %cd notebooks
4 from install import *
5 install_requirements()
```

```
[12] 1 # hide
2 from utils import *
3 setup_chapter()
```

The Dataset

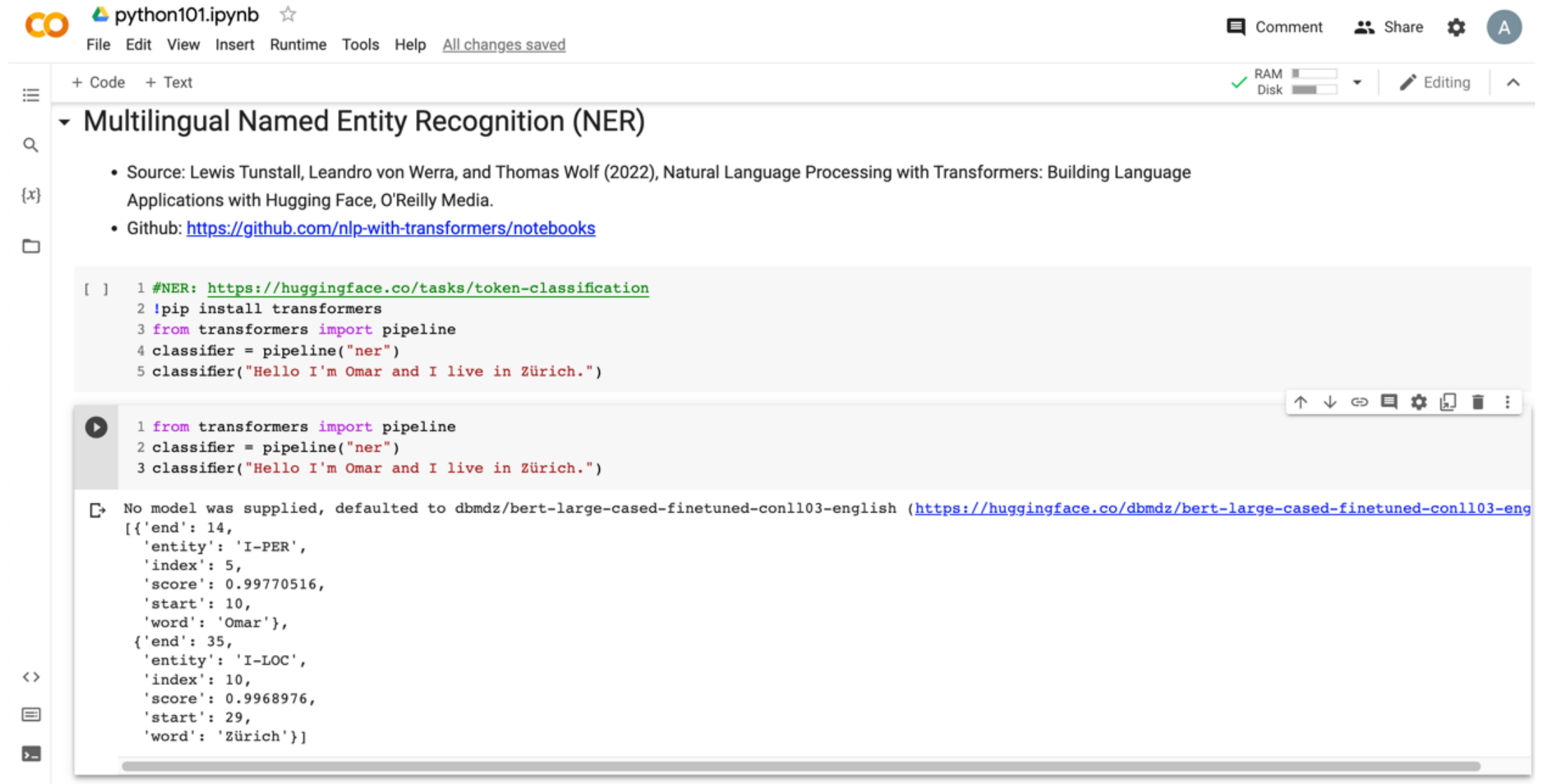
```
[13] 1 from datasets import list_datasets
2 all_datasets = list_datasets()
3 print(f"There are {len(all_datasets)} datasets currently available on the Hub")
4 print(f"The first 10 are: {all_datasets[:10]}")
```

There are 3783 datasets currently available on the Hub
The first 10 are: ['acronym_identification', 'ade_corpus_v2', 'adversarial_qa',

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Editing

▾ Multilingual Named Entity Recognition (NER)

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
[ ] 1 #NER: https://huggingface.co/tasks/token-classification
    2 !pip install transformers
    3 from transformers import pipeline
    4 classifier = pipeline("ner")
    5 classifier("Hello I'm Omar and I live in Zürich.")
```

```
▶ 1 from transformers import pipeline
   2 classifier = pipeline("ner")
   3 classifier("Hello I'm Omar and I live in Zürich.")
```

```
↳ No model was supplied, defaulted to dbmdz/bert-large-cased-finetuned-conll103-english (https://huggingface.co/dbmdz/bert-large-cased-finetuned-conll103-eng)
[{'end': 14,
  'entity': 'I-PER',
  'index': 5,
  'score': 0.99770516,
  'start': 10,
  'word': 'Omar'},
 {'end': 35,
  'entity': 'I-LOC',
  'index': 10,
  'score': 0.9968976,
  'start': 29,
  'word': 'Zürich'}]
```

<https://tinyurl.com/aintpuppython101>

Named Entity Recognition (NER)

```
from transformers import pipeline
import pandas as pd
classifier = pipeline("ner")
text = "My name is Michael and I live in Berkeley, California."
outputs = classifier(text)
pd.DataFrame(outputs)
```

	entity	score	index	word	start	end
0	I-PER	0.998874	4	Michael	11	18
1	I-LOC	0.997050	9	Berkeley	33	41
2	I-LOC	0.999170	11	California	43	53

Summary

- **Multilingual Named Entity Recognition (NER)**
- **Text Similarity and Clustering**

References

- Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Denis Rothman (2021), Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more, Packt Publishing.
- Savaş Yıldırım and Meysam Asgari-Chenaghlu (2021), Mastering Transformers: Build state-of-the-art models from scratch with advanced natural language processing techniques, Packt Publishing.
- Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta (2020), Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems, O'Reilly Media.
- Jing Li, Aixin Sun, Jianglei Han, and Chenliang LI (2022). "A survey on deep learning for named entity recognition." IEEE Transactions on Knowledge and Data Engineering 34, no. 1 (2022): 50-70.
- Zara Nasar, Syed Waqar Jaffry, and Muhammad Kamran Malik (2022). "Named entity recognition and relation extraction: State-of-the-art." ACM Computing Surveys (CSUR) 54, no. 1 (2022): 1-39.
- Pan Liu, Yanming Guo, Fenglei Wang, and Guohui Li (2022). "Chinese named entity recognition: The state of the art." Neurocomputing 473 (2022): 37-53.
- Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson.
- Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress.
- Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda (2018), Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning, O'Reilly.
- Charu C. Aggarwal (2018), Machine Learning for Text, Springer.
- Gabe Ignatow and Rada F. Mihalcea (2017), An Introduction to Text Mining: Research Design, Data Collection, and Analysis, SAGE Publications.
- Rajesh Arumugam (2018), Hands-On Natural Language Processing with Python: A practical guide to applying deep learning architectures to your NLP applications, Packt.
- Jake VanderPlas (2016), Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly Media.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." arXiv preprint arXiv:1810.04805.
- The Super Duper NLP Repo, <https://notebooks.quantumstat.com/>
- Jay Alammar (2018), The Illustrated Transformer, <http://jalammar.github.io/illustrated-transformer/>
- Jay Alammar (2019), A Visual Guide to Using BERT for the First Time, <http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/>
- NLP with Transformer, <https://github.com/nlp-with-transformers/notebooks>
- Min-Yuh Day (2022), Python 101, <https://tinyurl.com/aintpupython101>