AACL

Efficient and Robust Knowledge Graph Construction

Ningyu Zhang, Zhejiang University

Tao Gui, Fudan University

Guoshun Nan, Beijing University of Posts and Telecommunications

Schedule

Introduction and Application

- Named Entity Recognition
- Relation Extraction
- Knowledge Graph Construction

By: Guoshun Nan

Efficient KG Construction

- Data Efficiency
- Model Efficiency
- Inference Efficiency

By : Tao Gui

Robust KG Construction

- Robustness Problem Discovery
- Data Correction
- Robust Model Learning

By : Ningyu Zhang

Instructors

Ningyu Zhang

Zhejiang University

Tao Gui

Fudan University

Guoshun Nan

Beijing University of Posts and Telecommunications

Efficient and Robust
Knowledge Graph Construction

Github: https://github.com/NLP-Tutorials/AACL-IJCNLP2022-KGC-Tutorial

Questions: zhangningyu@zju.edu.cn, tgui@fudan.edu.cn, nanguo2021@bupt.edu.cn

Part 1

Named Entity Recognition

(NER)

Named Entity Recognition(NER)

What is Named Entity Recognition(NER)?

Recognize mentions of rigid designators from text belonging to predefined semantic types, such as person, location, organization etc.

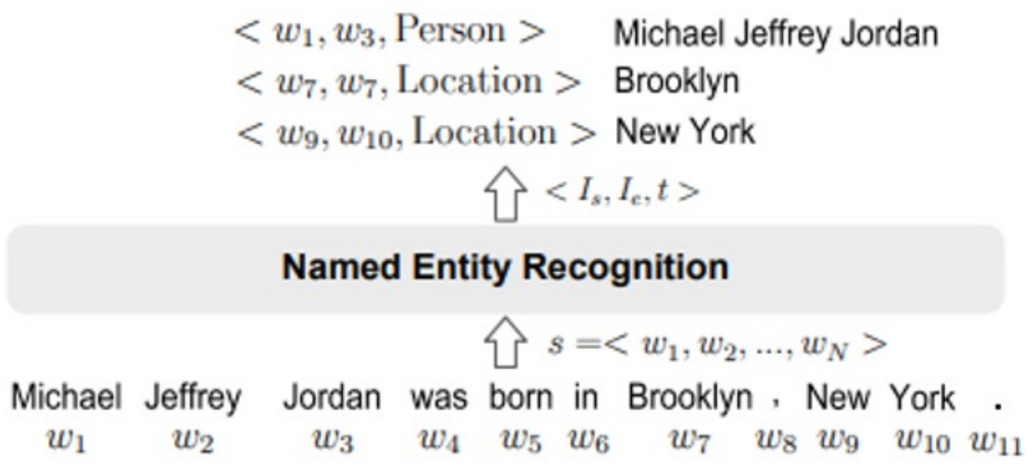


Fig.1. An illustration of the named entity recognition task.

Named Entity Recognition(NER)

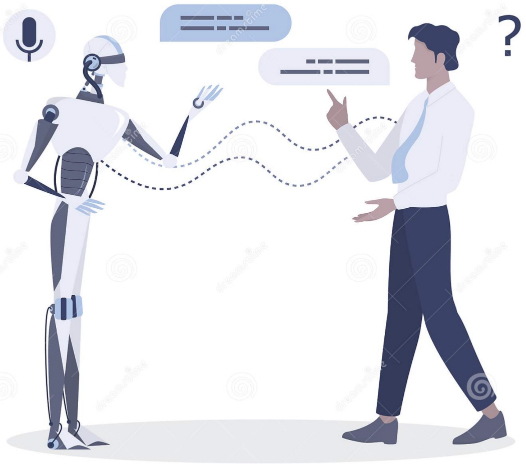
Downstream Tasks

NER is not a standalone tool for information extraction(IE).

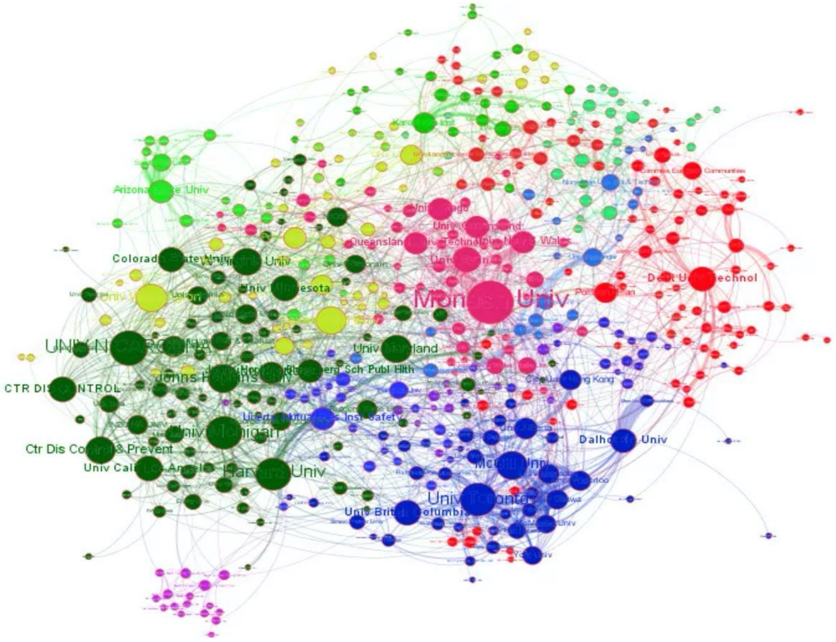
NER plays an essential role in a variety of natural language processing(NLP) applications.

- ❑ machine translation

- ❑ question answering



- ❑ knowledge base construction



Scope: Development process of NER

Ruled-based approaches: Rely on hand-crafted rules

- Based on domain-specific knowledge and cannot be transferred to other d

Feature-based approaches: NER is cast to a multi-class classification or sequence labeling tas

- Machine learning approaches need carefully designed features, which needs a lot of human effort.

Deep learning approaches:

- NER benefits from the non-linear transformation
- Deep learning saves significant effort on designing NER features
- Deep neural NER models can be trained in an end-to-end paradigm.

Flat NER

Flat NER：

Typical approach: BERT+LSTM+CRF

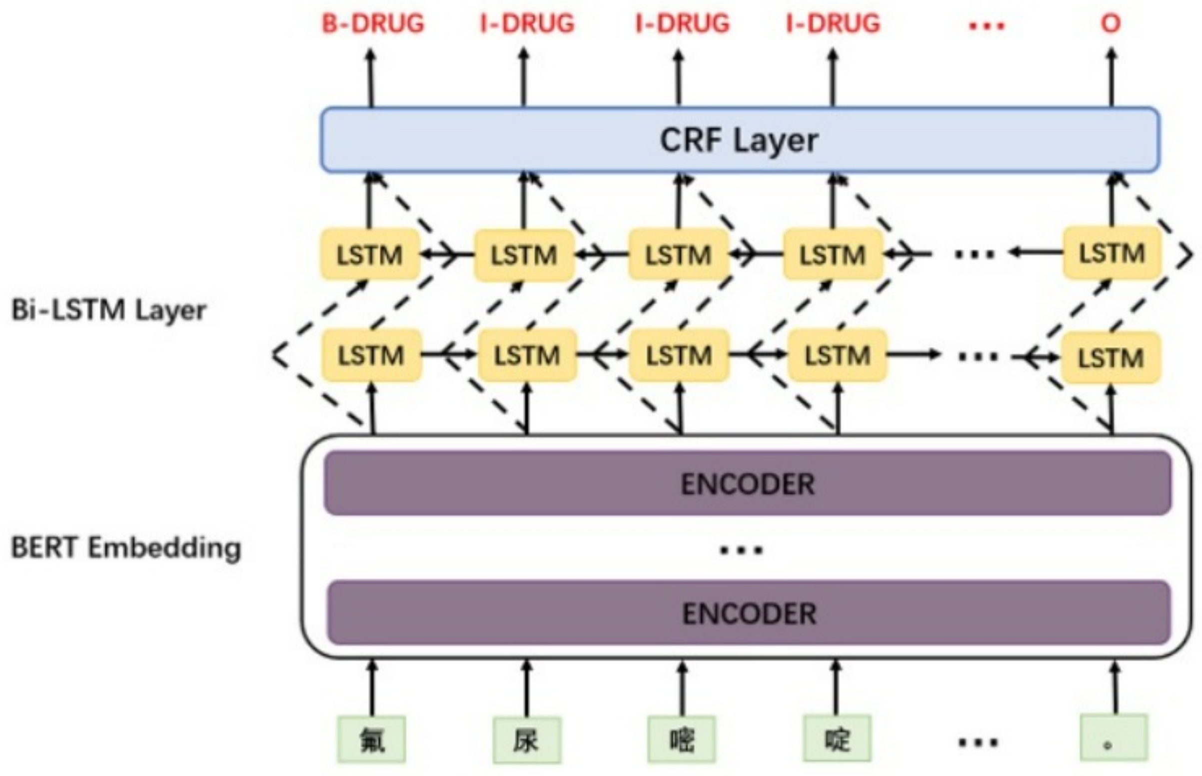


Fig 1: A brief overview of this kind of architecture

Nested NER

Nested NER : further deals with entities that can be nested with each other

One solution : Combine a flat module for outermost entities and a graph module for inner entities.

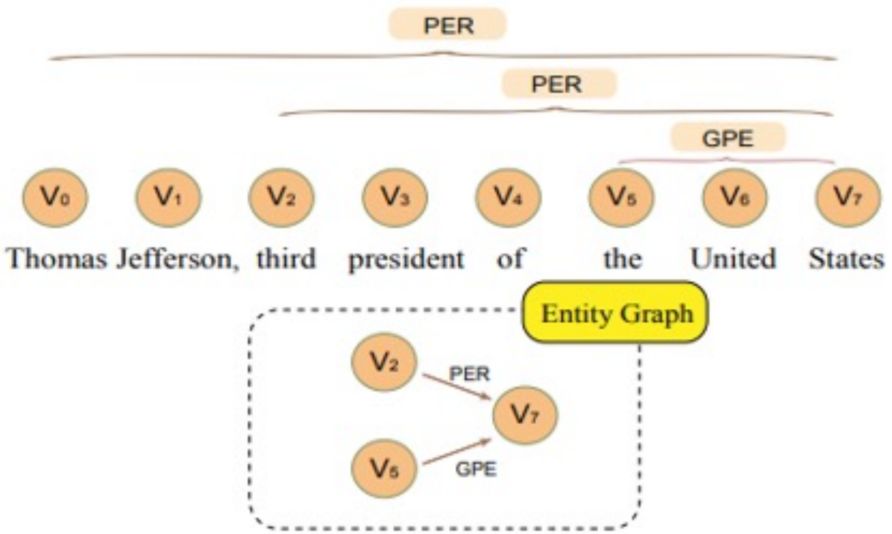


Figure 1: An example of nested named entity mentions.

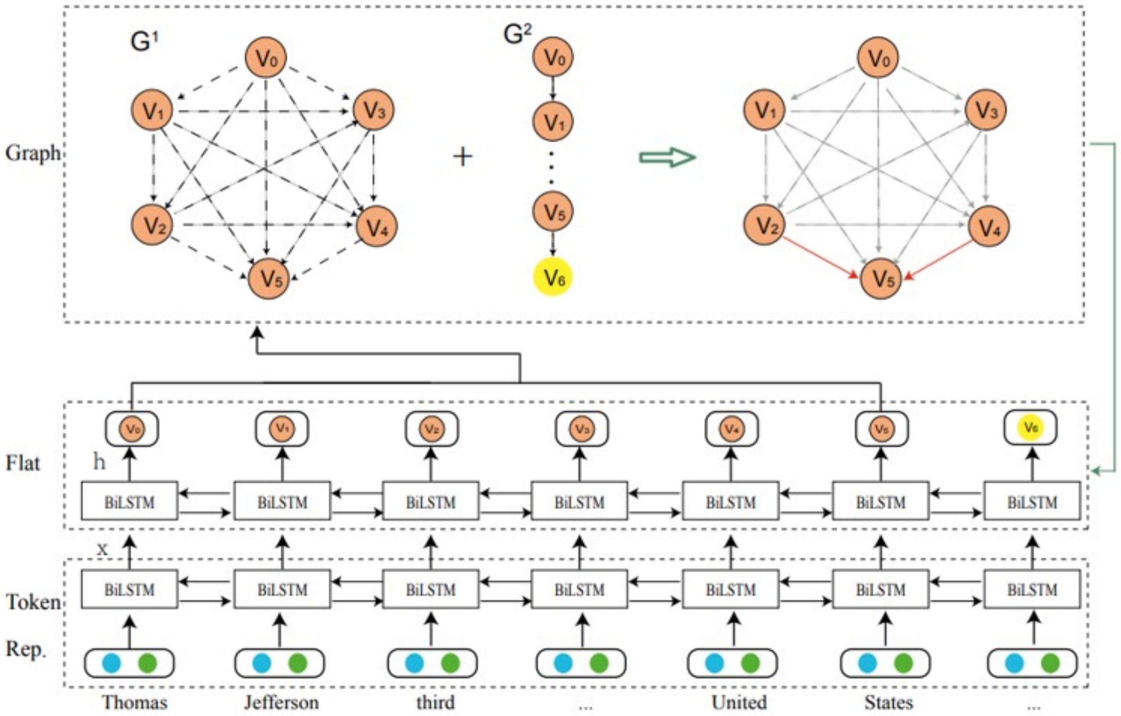


Figure 2:The framework of the BiFlaG model.

Bipartite Flat-Graph Network for Nested Named Entity Recognition Ying Luo et.al (ACL 2020)

Joint Flat NER and Nested NER

Solutions : Formulate the NER subtasks as an entity span sequence generation task, solved by a unified seq2seq framework.

The following is examples of three kinds of NER subtasks and their mainstream solutions respectively:

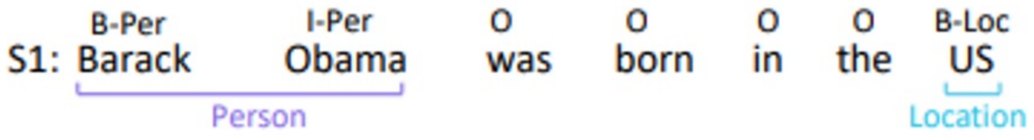


Figure 1: Sequenece labeling for flat NER

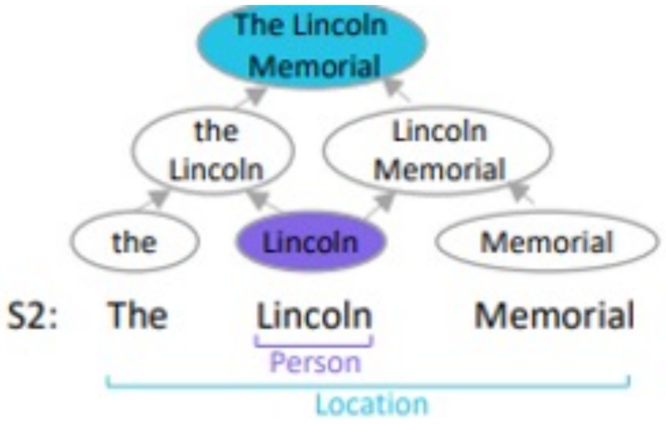


Figure 2: Span-based classification for nested NER

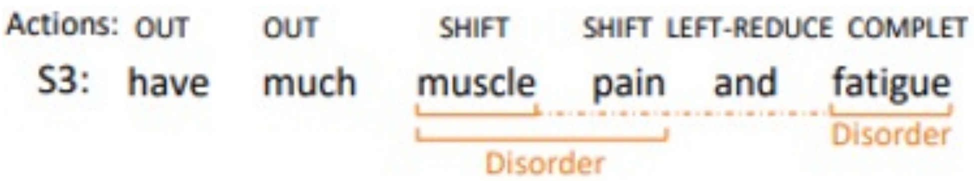


Figure 3: Transition-based method for discontinuous NER

Joint Flat NER and Nested NER

Joint Flat and Nested NER

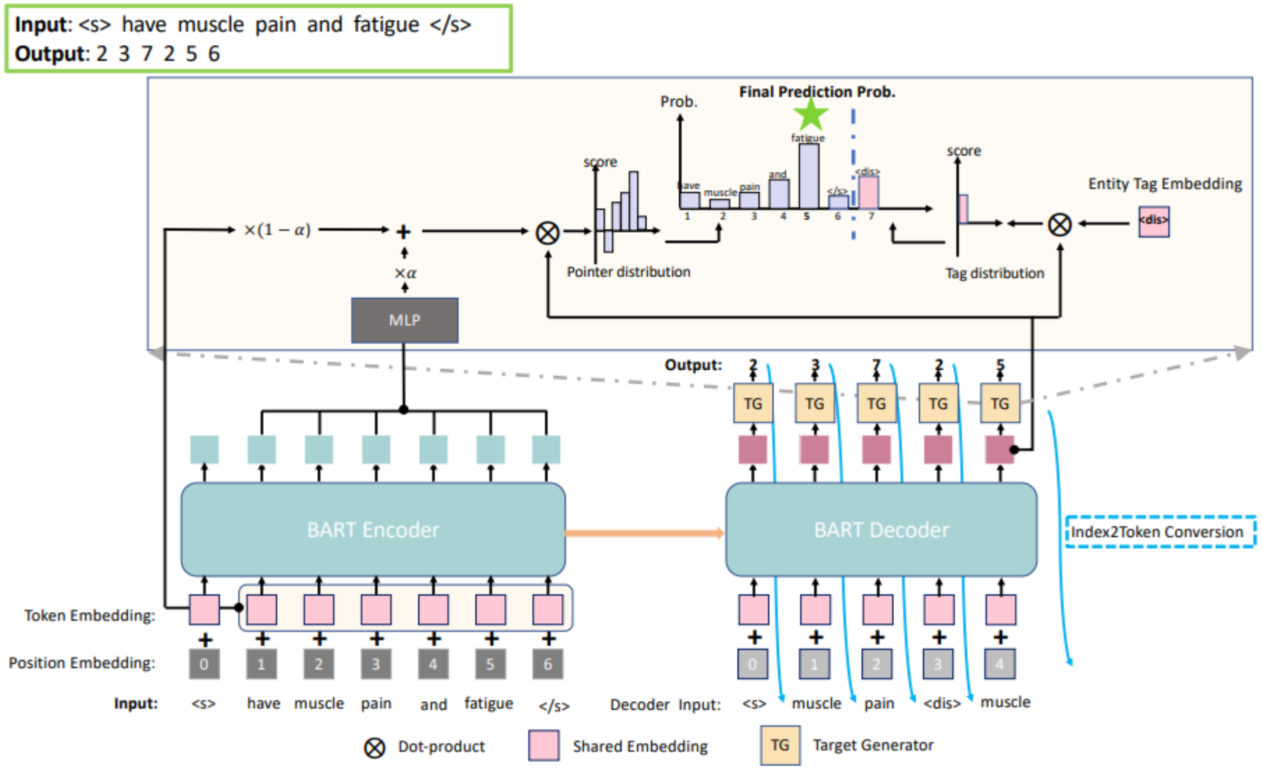


Figure 1:Model structure used in this method

A Unified Generative Framework for Various NER Subtasks Hang Yan et.al (ACL 2021)

Joint NER and Nested NER

Joint Flat and Nested NER :

Problems: Sequence labeling models for flat NER are unsuitable for nested NER.

Solutions: Formulate the NER subtasks as a machine reading comprehension(MRC) task

for each entity $X_{start,end}$ with a label y

triple (q, $X_{start,end}$, X)

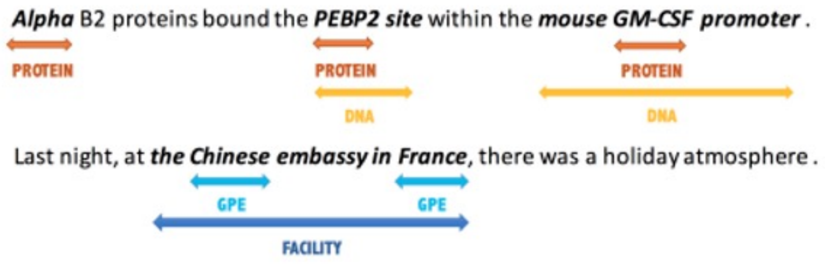


Figure 1: Examples for nested entities from GENIA and ACE04 corpora

Entity	Natural Language Question
Location	Find locations in the text, including non-geographical locations, mountain ranges and bodies of water.
Facility	Find facilities in the text, including buildings, airports, highways and bridges.
Organization	Find organizations in the text, including companies, agencies and institutions.

Table1:Examples for transforming different entity categories to question queries.

A Unified MRC Framework for Named Entity Recognition Xiaoya Li et.al (ACL 2020)

Chinese NER

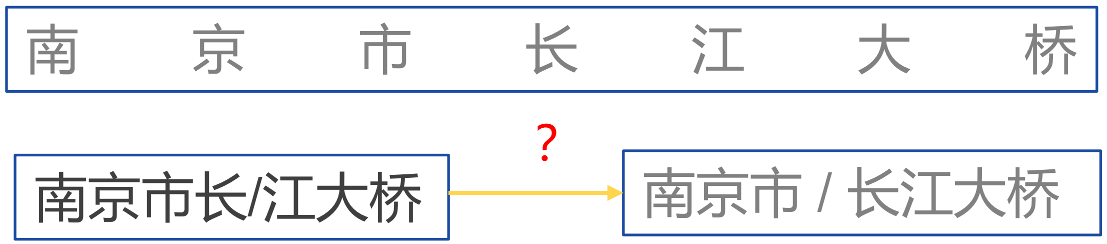


Fig 1: An example of Chinese NER

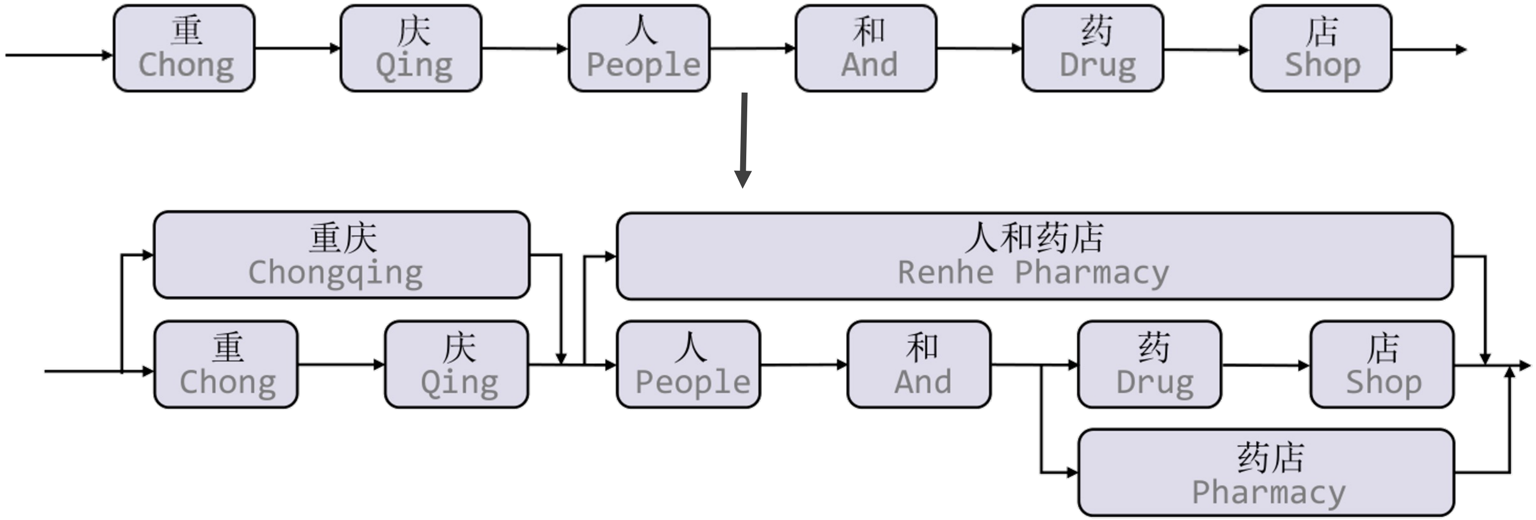


Fig 2: Lattice structure in Chinese NER

Chinese NER

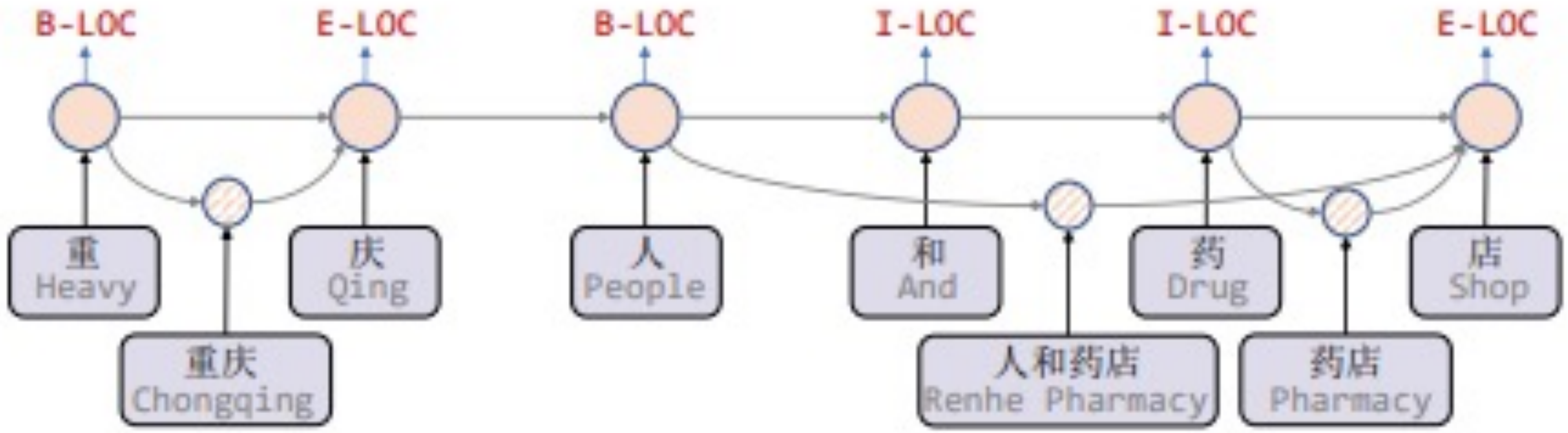


Fig 1: Lattice LSTM

FLAT: Chinese NER Using Flat-Lattice Transformer Xiaonan Li et.al (ACL 2020)

NER Datasets

Trend: Multi-lingual&multi-modal&mult-domain

Dataset:

Entity Type	Train	Valid.	Test	All
PATIENT_ID	3240	1276	2005	6521
PERSON_NAME	349	188	318	855
AGE	682	361	582	1625
GENDER	542	277	462	1281
OCCUPATION	205	132	173	510
LOCATION	5398	2737	4441	12576
ORGANIZATION	1137	551	771	2459
SYMPTOM&DISEASE	1439	766	1136	3341
TRANSPORTATION	226	87	193	506
DATE	2549	1103	1654	5306
# Entities in total	15767	7478	11735	34984
# Sentences in total	5027	2000	3000	10027

Table 1: Statistics of the first manually annotated Vietnamese dataset in the COVID-19 domain.

Dataset	# Train	# Dev	# Test	# Total	Language	Structure	Modality
MSRA	46,364	-	4,365	50,729	Chinese	Flat	Text
OntoNotes	15,724	4301	4,346	24,371	Chinese	Flat	Text
Weibo NER	1,350	271	270	1,891	Chinese	Flat	Text
Resume	3,821	463	477	4,761	Chinese	Flat	Text
GENIA	15,022	1,669	1,854	18,545	English	Nested	Text
JNLPBA	20,546	-	4,260	24,806	English	Nested	Text
ACE-2004	6,198	742	809	7,749	English	Nested	Text
ACE-2005	7,285	968	1,058	9,311	English	Nested	Text
Twitter-2015	4,000	1,000	3,257	8,257	English	Flat	Text + Image
Twitter-2017	3,373	723	723	4,819	English	Flat	Text + Image
CNERTA	34,102	4,440	4,445	42,987	Chinese	Nested	Text + Speech

Table 2:A comparison between CNERTA and other existing widely-used NER datasets.

COVID-19 Named Entity Recognition for Vietnamese Thinh Hung Truong et.al (NAACL 2021)

A Large-Scale Chinese Multimodal NER Dataset with Speech Clues Dianbo Sui et.al

Zero-shot/Few-shot NER

Few-shot NER exploits only a handful of annotations to identify and classify named entity mentions.

Previous work: Prototypical networks show superior performance on few-shot NER.

One problem: fail to differentiate rich semantics in other-class words.

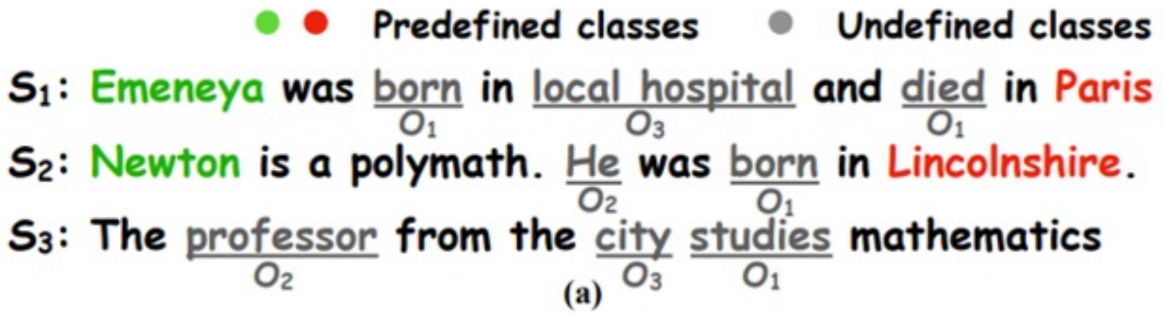


Figure (a): Examples for undefined classes

Learning from Miscellaneous Other-Class Words for Few-shot Named Entity Recognition Meihan Tong et.al (ACL 2021)

Zero-shot/Few-shot NER-MUCO

Follows are the architecture of one proposed MUCO model to solve the problem.

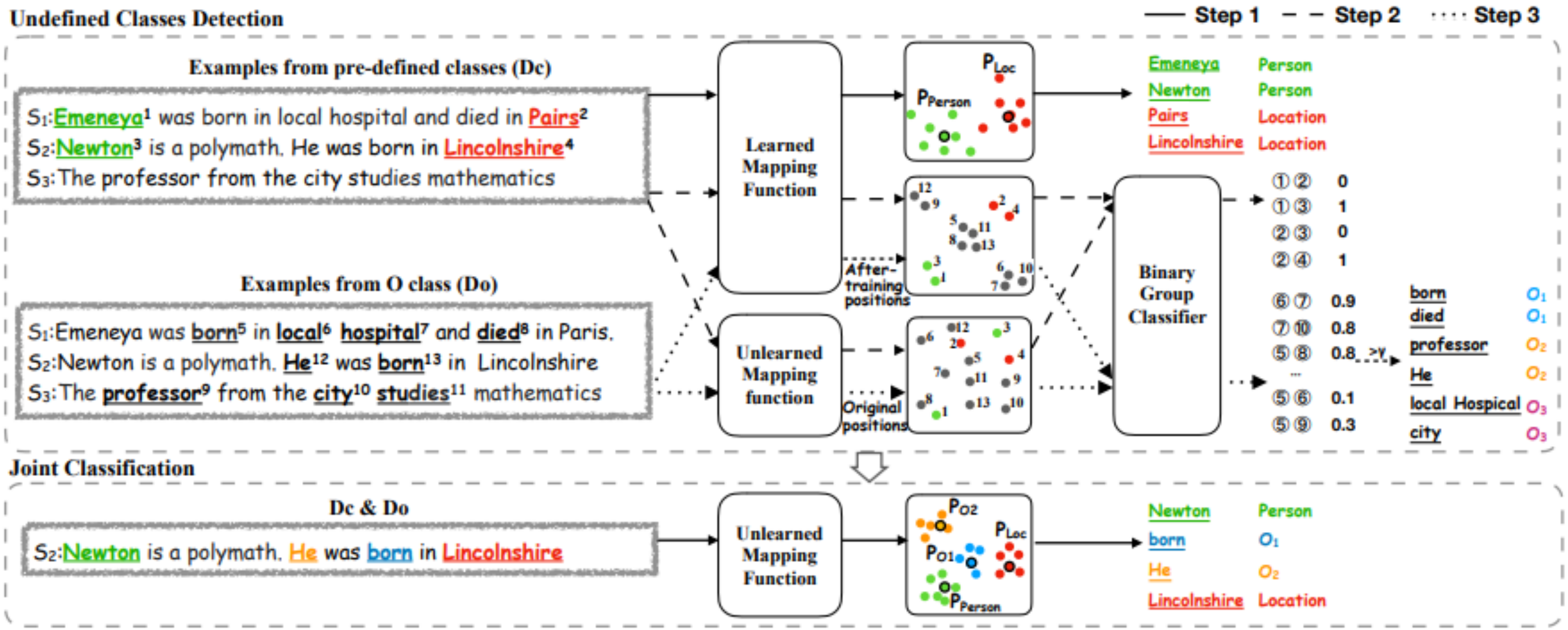


Figure 1: The architecture of the proposed MUCO model.

Zero-shot/Few-shot NER-Prompt tunning

Future direction: zero-shot/few-shot NER

Another method:

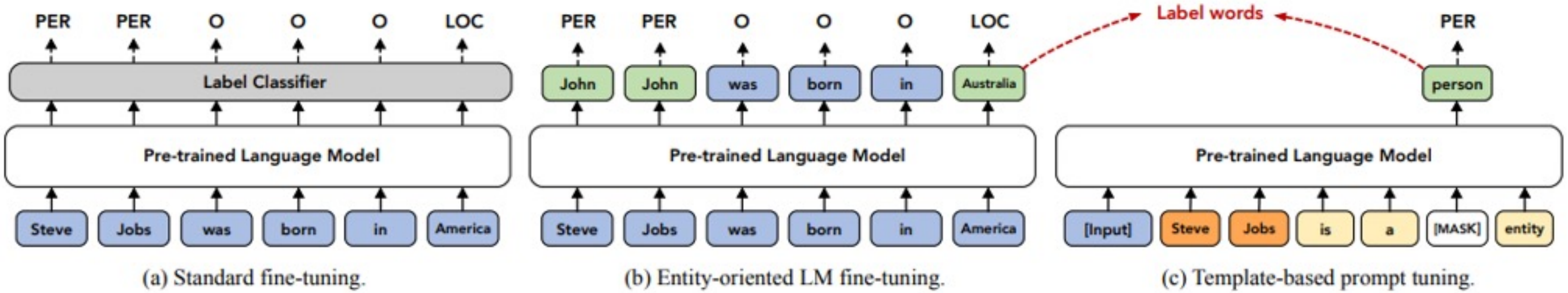


Figure 1: Comparison of different fine-tuning methods for NER.

Template-free Prompt Tuning for Few-shot NER Tao Gui et.al (NAACL 2022)

Low-Resource NER: Joint Labeling

Token-level labels are costly to annotate

One Method: Exploiting sentence-level labels, which are easy to obtain

Category: ELECTRONICS

Title:	Ốp	lưng	silicon	dẻo	Hàn	Quốc
Label:	B-PRODUCT	E-PRODUCT	S-MATERIAL	S-PATTERN	O	O
Translation:	*case*		*silicon*	*flexible*	*Korea*	

"... Korean flexible silicon case ..."

Category: HEALTH_BEAUTY

Title:	COMBO	Gôm	xịt	tóc	Tigi	Bed	Head
Label:	O	B-PRODUCT	I-PRODUCT	E-PRODUCT	B-BRAND	I-BRAND	E-BRAND
Translation:	*combo*		*hairspray*		*Tigi*	*Bed*	*Head*

"... Tigi Bed Head hairspray combo ..."

Figure 1: Examples of product titles with NER annotation in Vietnamese

Low-Resource NER: Data Augmentation

Another solution: Data augmentation

However, when applied to token-level tasks such as NER, data augmentation methods often suffer from token-label misalignment, which leads to unsatsifactory performance

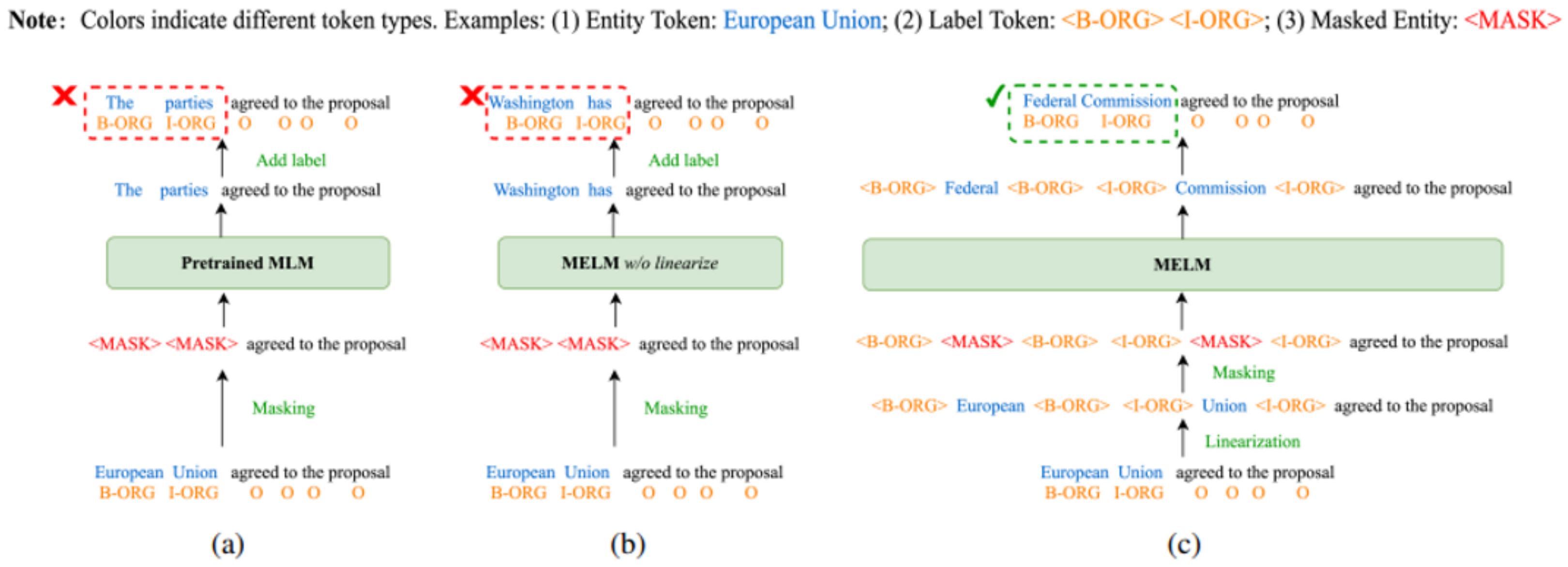


Figure 1: Comparison of different data augmentation methods, color printing is preferred.

MELM: Data Augmentation with Masked Entity Language Modeling for LowResource NER Ran Zhou et.al (ACL 2022)

Conclusion and Future work

Recent advances :

- Pretrained models
- Different frameworks
- New ideas, new methods

…

Open problems:

- Few-shot&zero-shot NER
- Unified framework for multi-task like flat NER and Nested NER
- Model generalization to multi-domain

…

Part 2

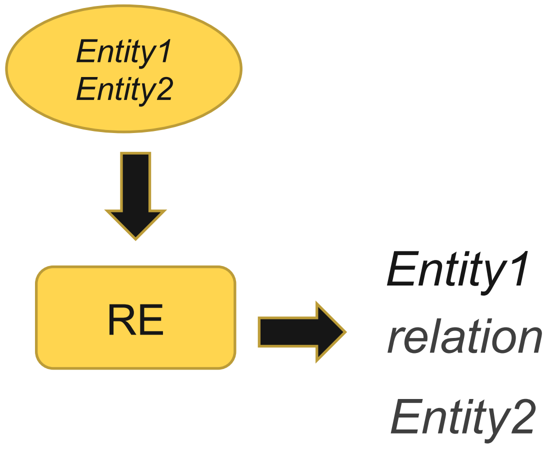
Relation Extraction

(RE)

Relation Extraction

What is Relation Extraction (RE) ?

Relation Extraction aims at extracting relational facts from plain text

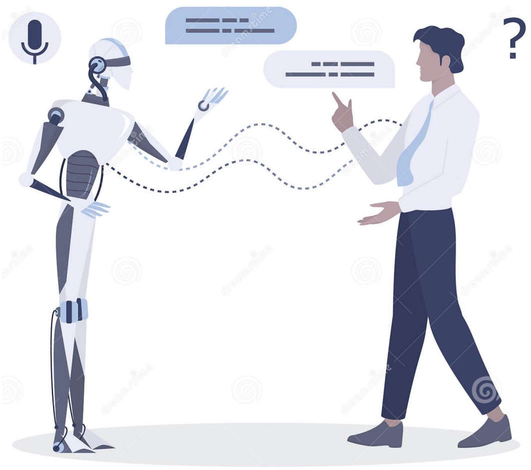


- ❑ **Examples**
  - ❑ Steve Jobs co-founded Apple (*Apple Inc.*, founded by, *SteveJobs*)
  - ❑ Hamilton made its debut in New York, USA (*USA*, contains, *New York*)
  - ❑ In 2002, Musk was excited to become the CEO of SpaceX. (*Musk*, works for, *SpaceX*)

Relation Extraction

Downstream Applications

- ❑ knowledge graph completion
- ❑ search engine
- ❑ question answering



Relation Extraction

AACL IJCNLP 2022

How to do relation extraction

Relation Classification/Open Relationship Extraction: Single sentence relation extraction

- Text classification model, dependency tree,...

Distant supervise: Pattern-based relation extraction

- Hand-Written Patterns, Semi-Supervised Learning,...

Document-level RE: Pre-trained model

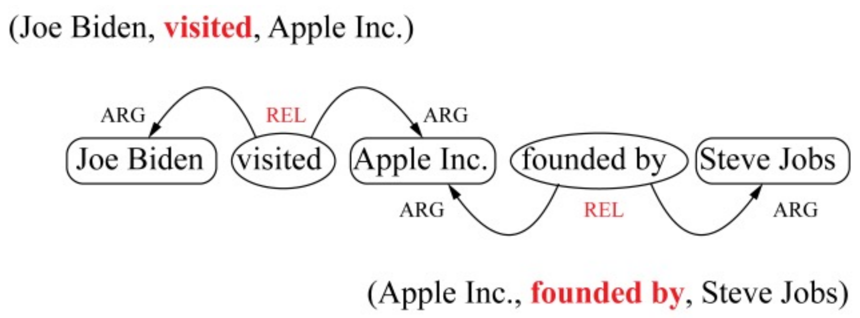
- Bert, Roberta, ...

Future Work : Low-shot / continual Learning

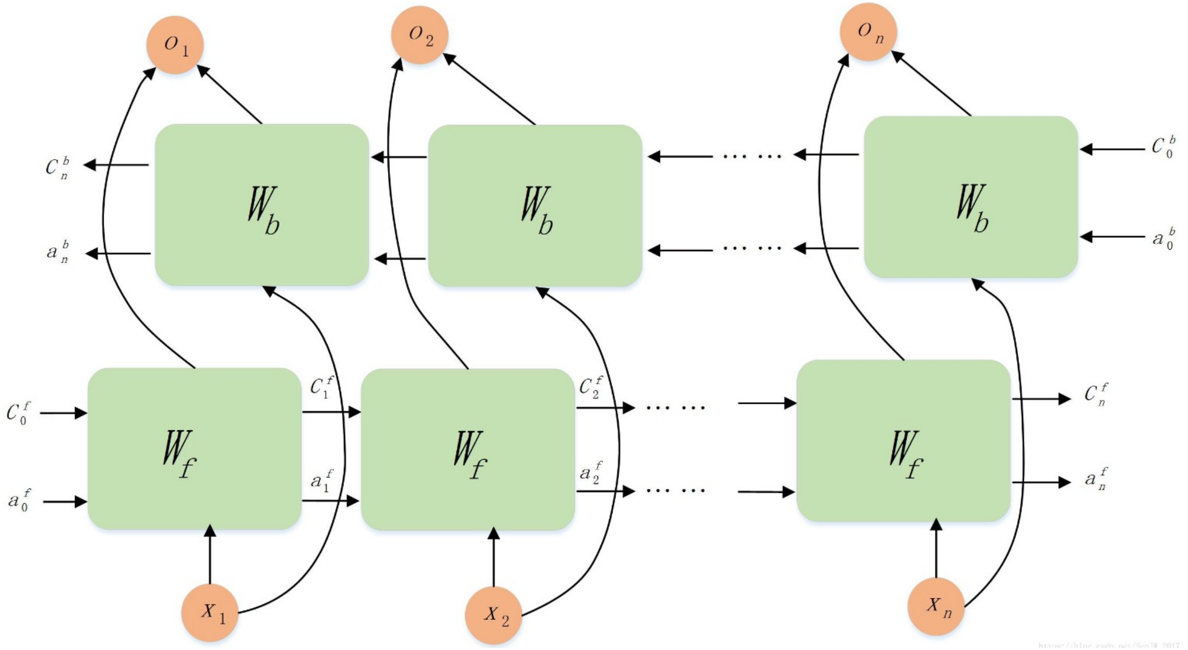
Relation Extraction

- Open Relation Extraction : extracting structured relation representations from text sentences .

Example :



Model :

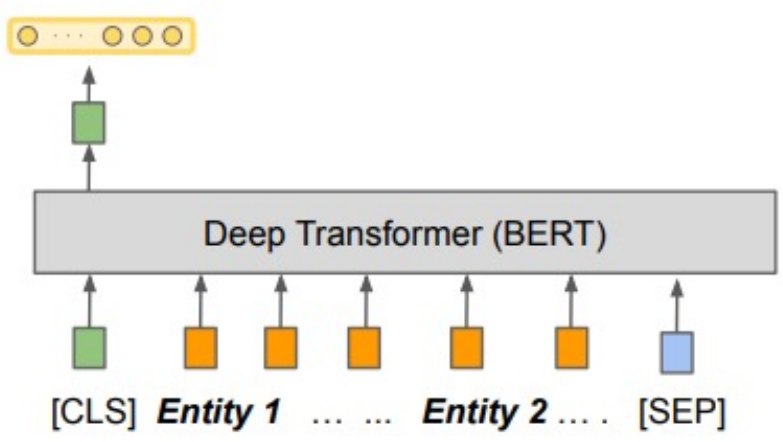


Explore BiLSTM-Based Models for Open Relation Extraction Tao Ni et.al (ACL2021)

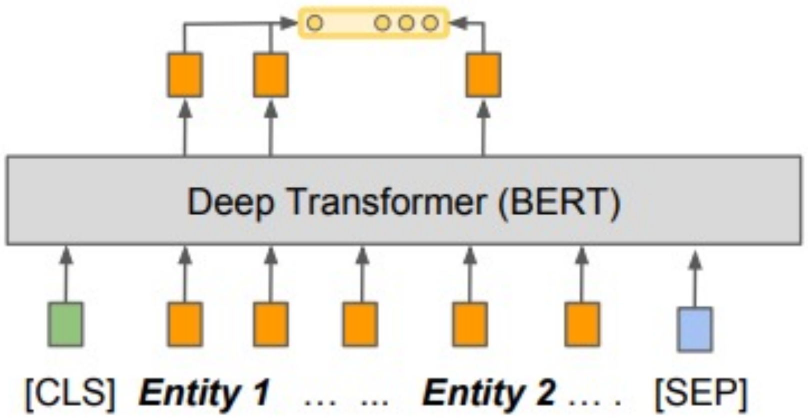
Relation Extraction

- Single sentence classification : Two entities in the same sentence.

  - General text classification model + position information (Pretrain model)



Baseline : (General text classification model)

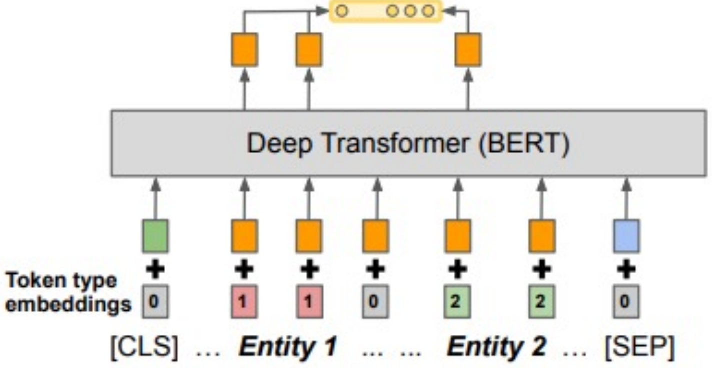


Mention pooling (max pooling)

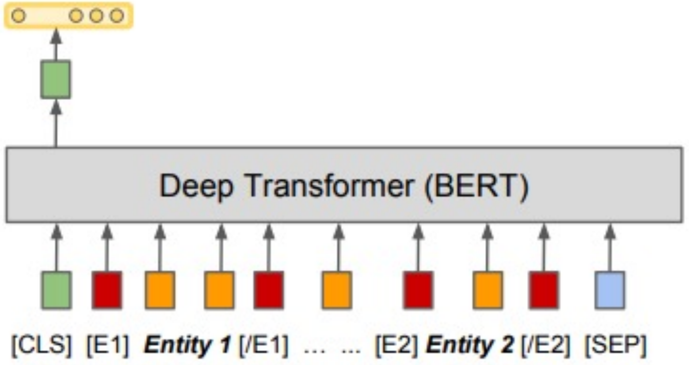
Matching the Blanks: Distributional Similarity for Relation Learning Tao Ni et.al (ACL2019)

Relation Extraction

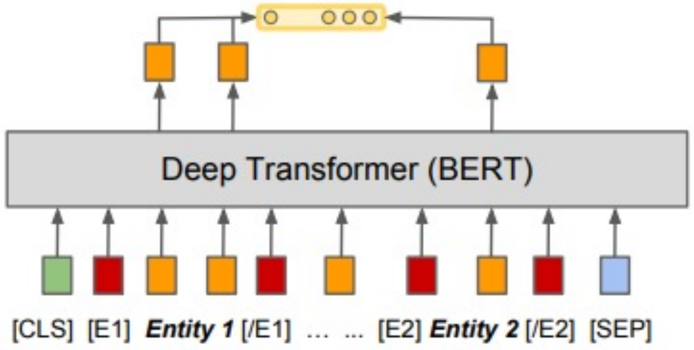
- Single sentence classification : Two entities in the same sentence.



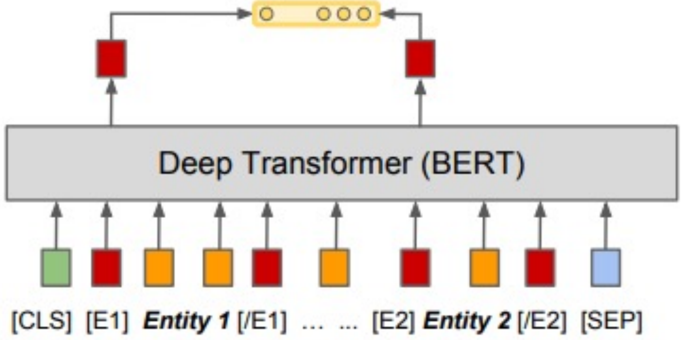
Position Embedding + Word



Position Embedding + Entity Markers



Entity embedding mention pooling



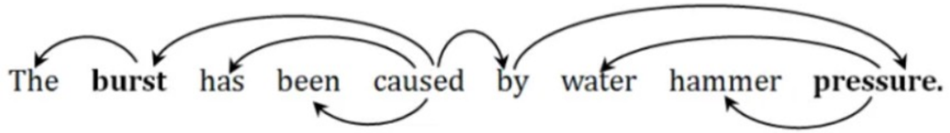
Token embedding mention pooling

Relation Extraction

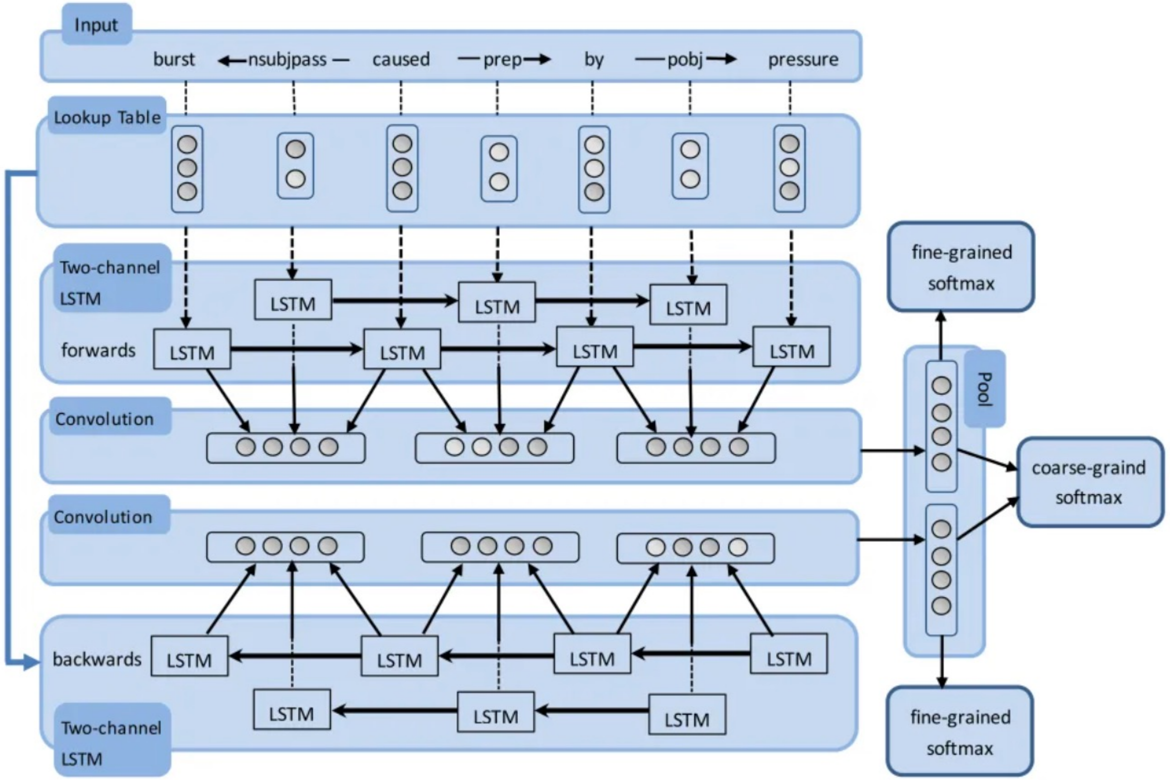
❑ Relation classification – dependency tree

Sentence --> Syntax dependency tree --> SDP

(Shortest Dependency Path)



Realtion	Shortest Dependency Path
Cause-Effect(e_2,e_1)	burst ←nsubjpass— caused —prep→ by —pobj→ pressure
Cause-Effect(e_1,e_2)	pressure ←pobj— by ←prep— caused —nsubjpass→ burst



Bidirectional recurrent convolutional neural network for relation classification Rui Cai et.al (ACL2016)

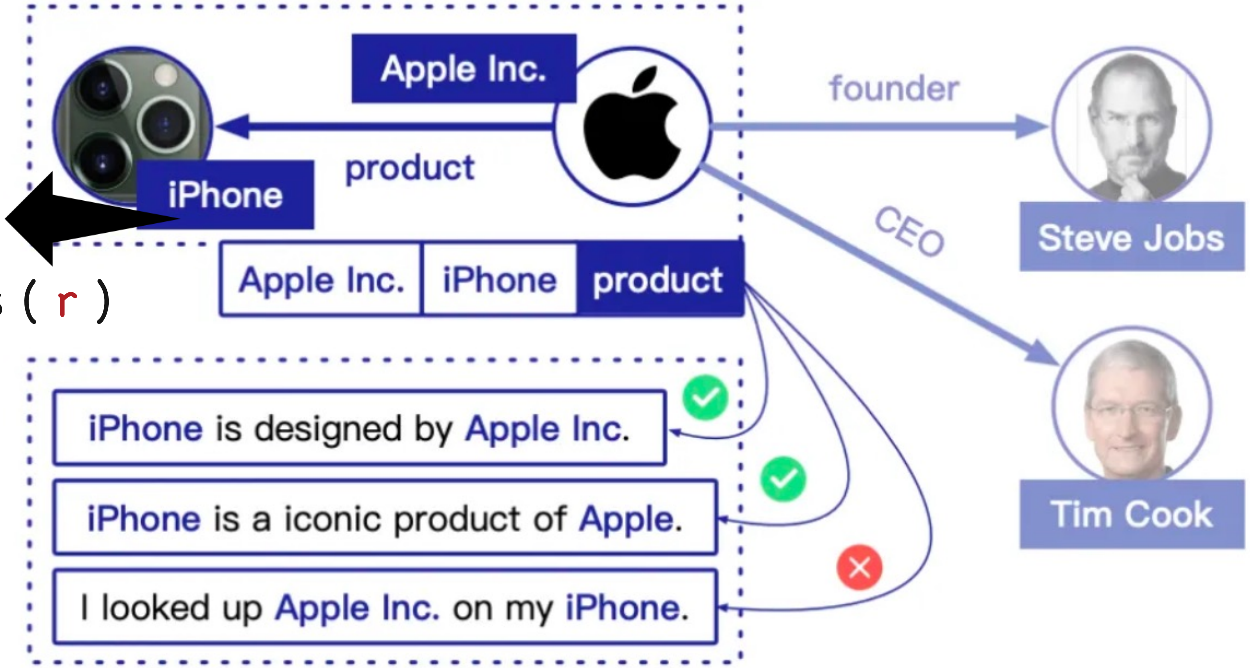
Relation Extraction

- Distant Supervise :

Use for : (Data generation / Relation extraction)

- Two entities in the same sentence (e1 , e2)
- A relation exists in the KG between two entities (r)

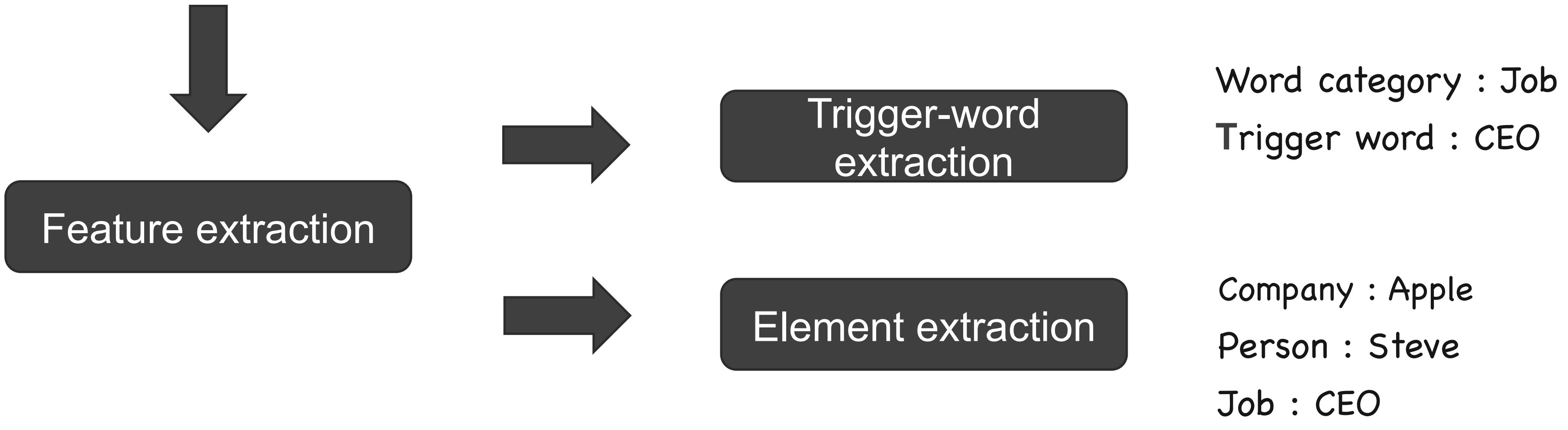
< e1 - r - e2 >



Relation Extraction

- Distant Supervise : *Distant supervision for relation extraction without labeled data*
  - Based on trigger words

In 1997 Steve was excited to become the CEO of Apple.



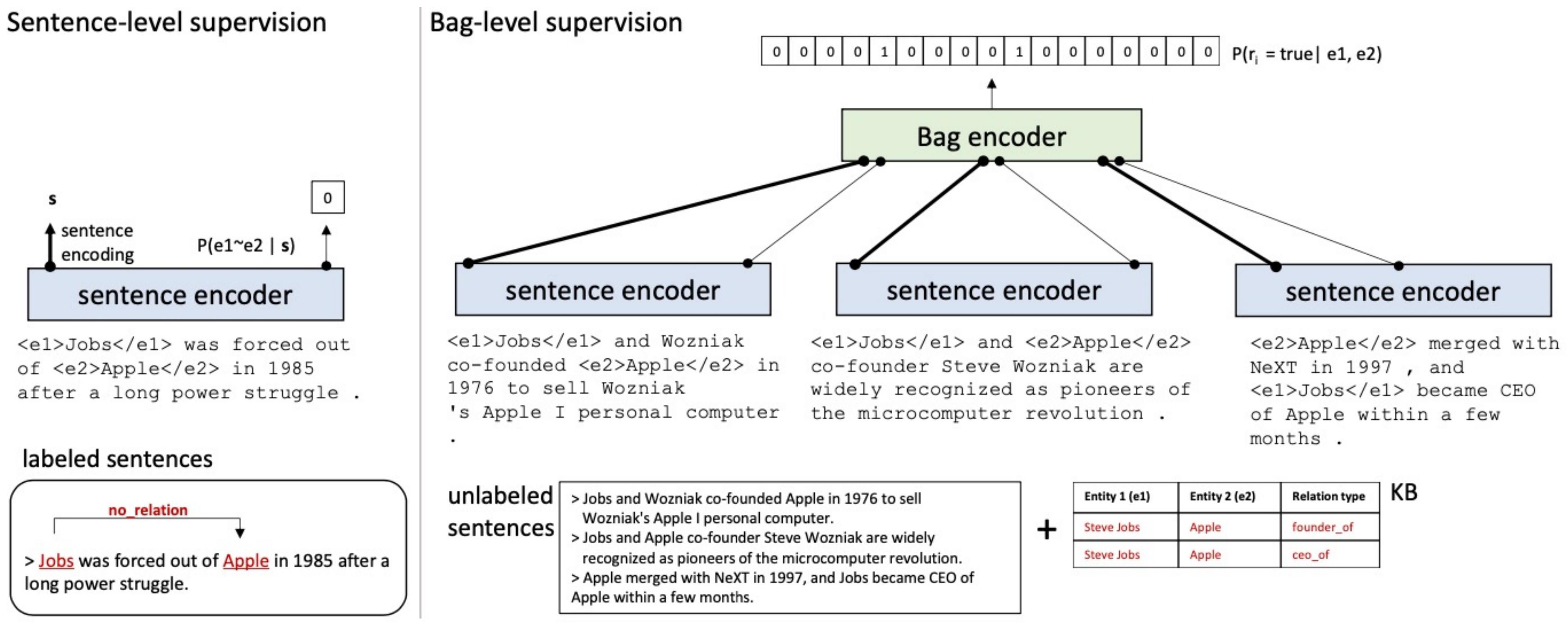
Relation Extraction

AACL IJCNLP 2022

- Distant Supervised RE :

Challenges : imbalanced data / noisy labels

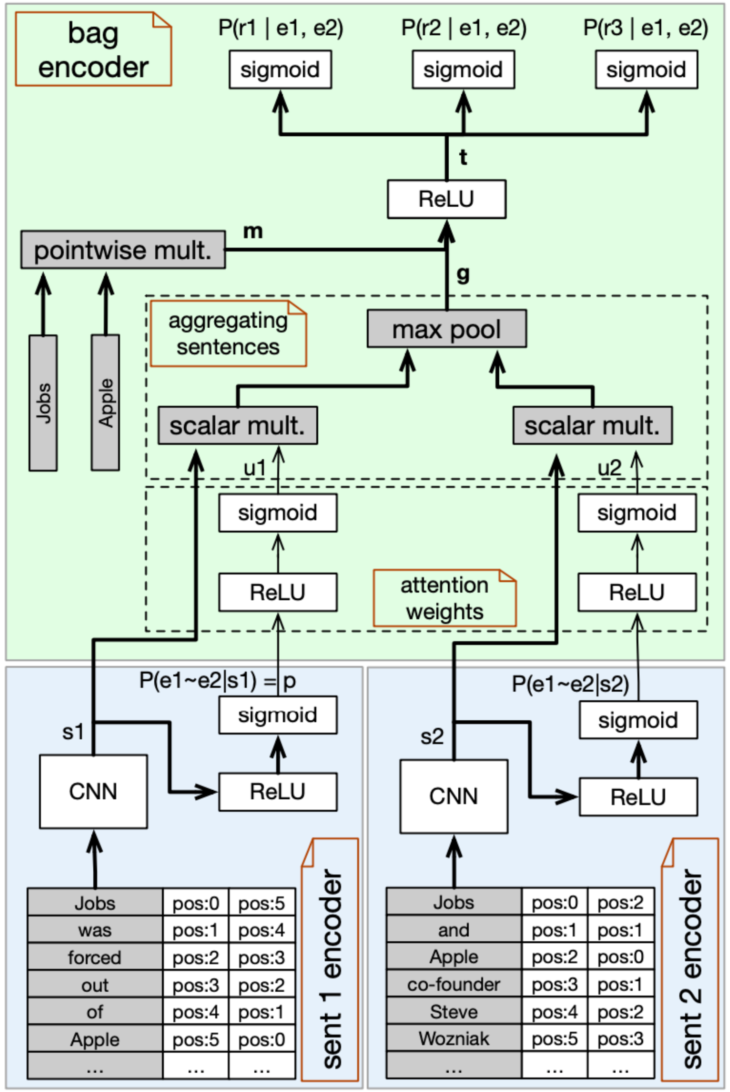
Approaches : combining the distant supervision data with an additional directly-supervised data



Relation Extraction

- Distant Supervised RE :
  - Optimized by direct supervision data
  - Optimized by weights calculated from the supervision data

direct
supervision
data



Combining Distant and Direct Supervision for Neural Relation Extraction Iz Beltagy et.al (ACL2020)

Relation Extraction

- Document - level relation extraction

Challenges :

- Entity disambiguation
- long distance dependency
- relation reasoning
- ...

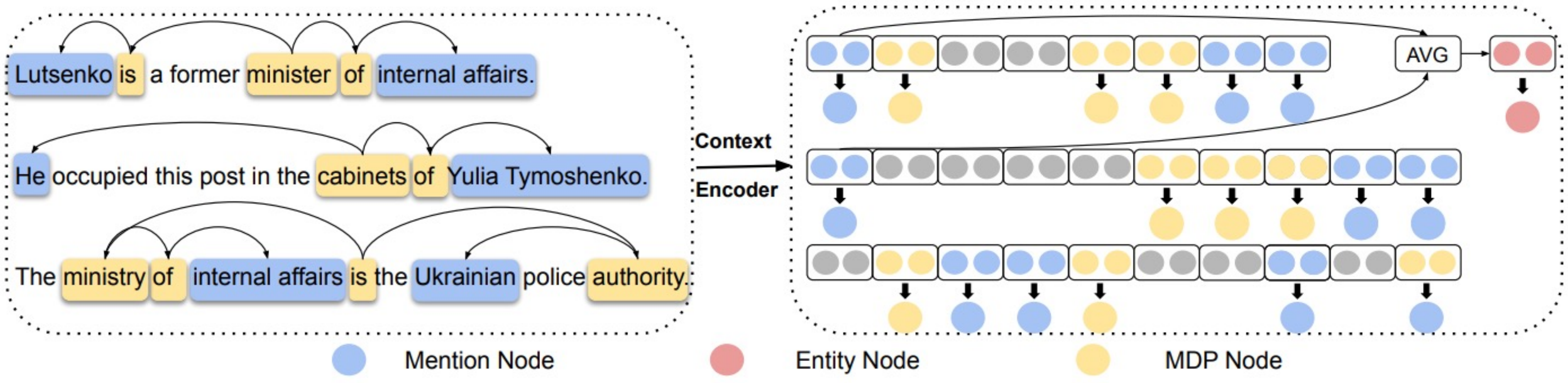
Here entity Lutsenko has two mentions: Lutsenko and He. Mentions corresponding to the same entity are highlighted with the same color

Relation Extraction

- Document-level relation extraction

Approach : treats the graph structure as a latent variable

Node Constructor

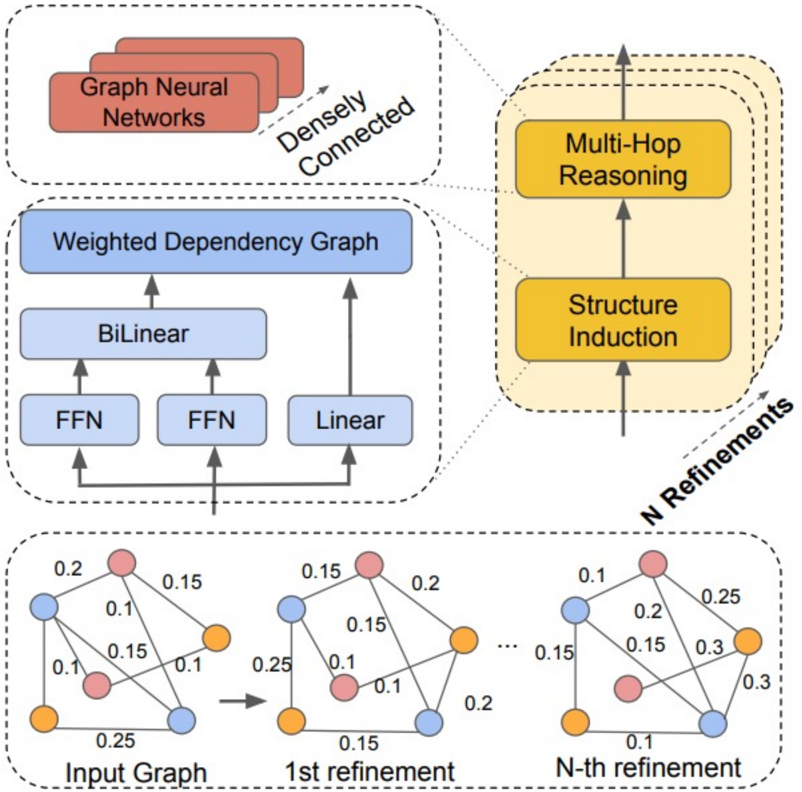


Reasoning with Latent Structure Refinement for Document-Level Relation Extraction Guoshun Nan et.al (ACL 2020)

Relation Extraction

❑ Document-level relation extraction

Dynamic Reasoner



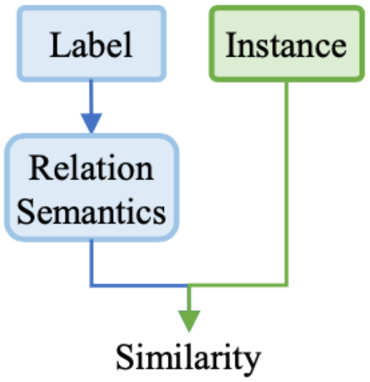
Model	Dev Ign $F1$	Dev $F1$	Dev Intra-$F1$	Dev Inter-$F1$	Test Ign $F1$	Test $F1$
CNN (Yao et al., 2019)	41.58	43.45	51.87*	37.58*	40.33	42.26
LSTM (Yao et al., 2019)	48.44	50.68	56.57*	41.47*	47.71	50.07
BiLSTM (Yao et al., 2019)	48.87	50.94	57.05*	43.49*	48.78	51.06
ContexAware (Yao et al., 2019)	**48.94**	51.09	56.74*	42.26*	48.40	50.70
GCNN ♣ (Sahu et al., 2019)	46.22	51.52	57.78	44.11	49.59	51.62
EoG ♣ (Christopoulou et al., 2019)	45.94	52.15	58.90	44.60	49.48	51.82
GAT ♣ (Veličković et al., 2018)	45.17	51.44	58.14	43.94	47.36	49.51
AGGCN ♣ (Guo et al., 2019a)	46.29	52.47	58.76	45.45	48.89	51.45
GloVe+LSR	48.82	**55.17**	**60.83**	**48.35**	**52.15**	**54.18**
BERT (Wang et al., 2019)	-	54.16	61.61*	47.15*	-	53.20
Two-Phase BERT (Wang et al., 2019)	-	54.42	61.80*	47.28*	-	53.92
BERT+LSR	**52.43**	**59.00**	**65.26**	**52.05**	**56.97**	**59.05**

Relation Extraction: Low shot learning

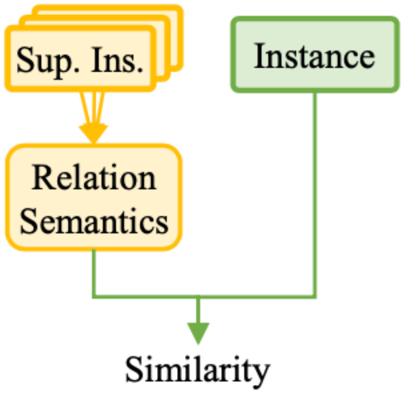
- Future work : Low-shot Learning

Low-shot relation extraction (RE) aims to recognize novel relations with very few or even no samples,

- Zero-shot



- Few-shot

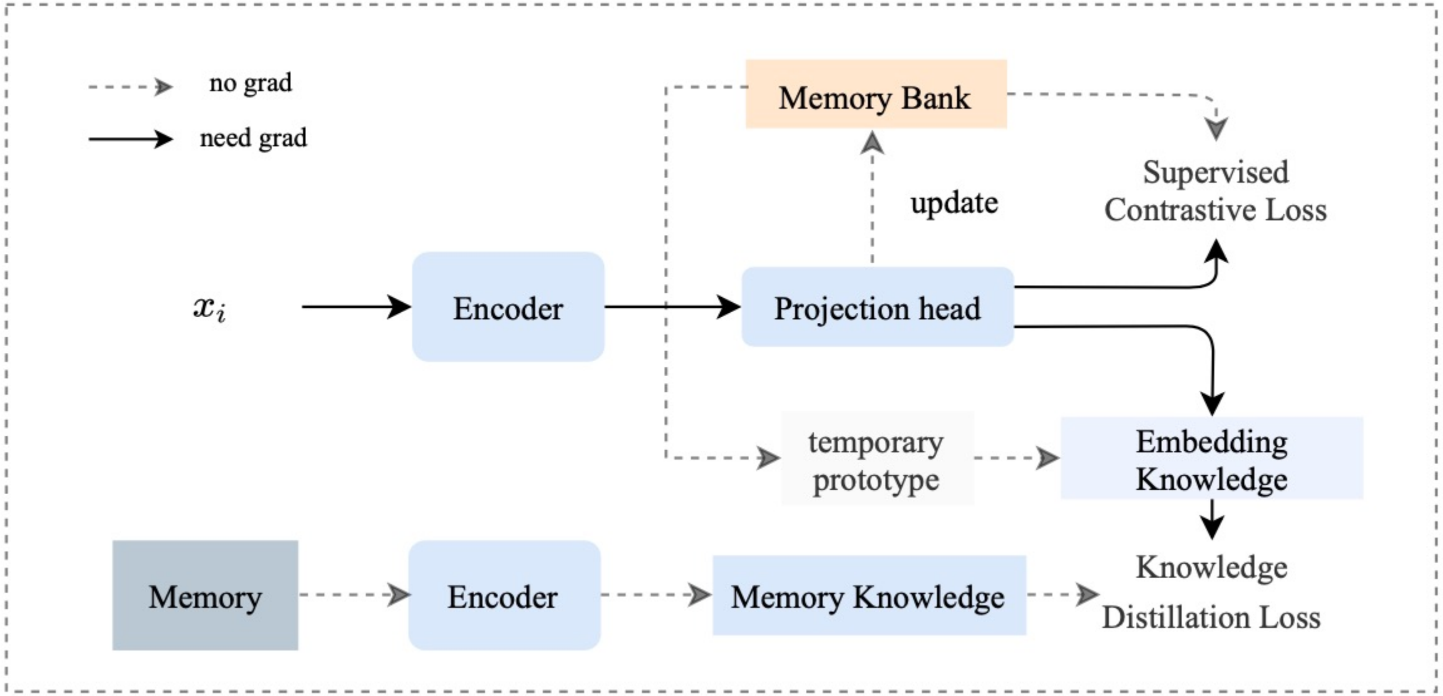


Pre-training to Match for Unified Low-shot Relation Extraction Fangchao Liu et.al (ACL 2022)

Relation Extraction

❑ Future work : continual learning

Continual relation extraction (CRE) aims to continuously train a model on data with new relations while avoiding forgetting old ones.



Consistent Representation Learning for Continual Relation Extraction Kang Zhao et.al (ACL 2022)

Conclusion and Future work

- Recent advances
  - The use of super large scale pre-training model
  - Significant progress has been made in simple relation extraction
- Future works
  - The performance of relation extraction in complex scenarios needs to be improved
  - Low-shot learning and continual learning

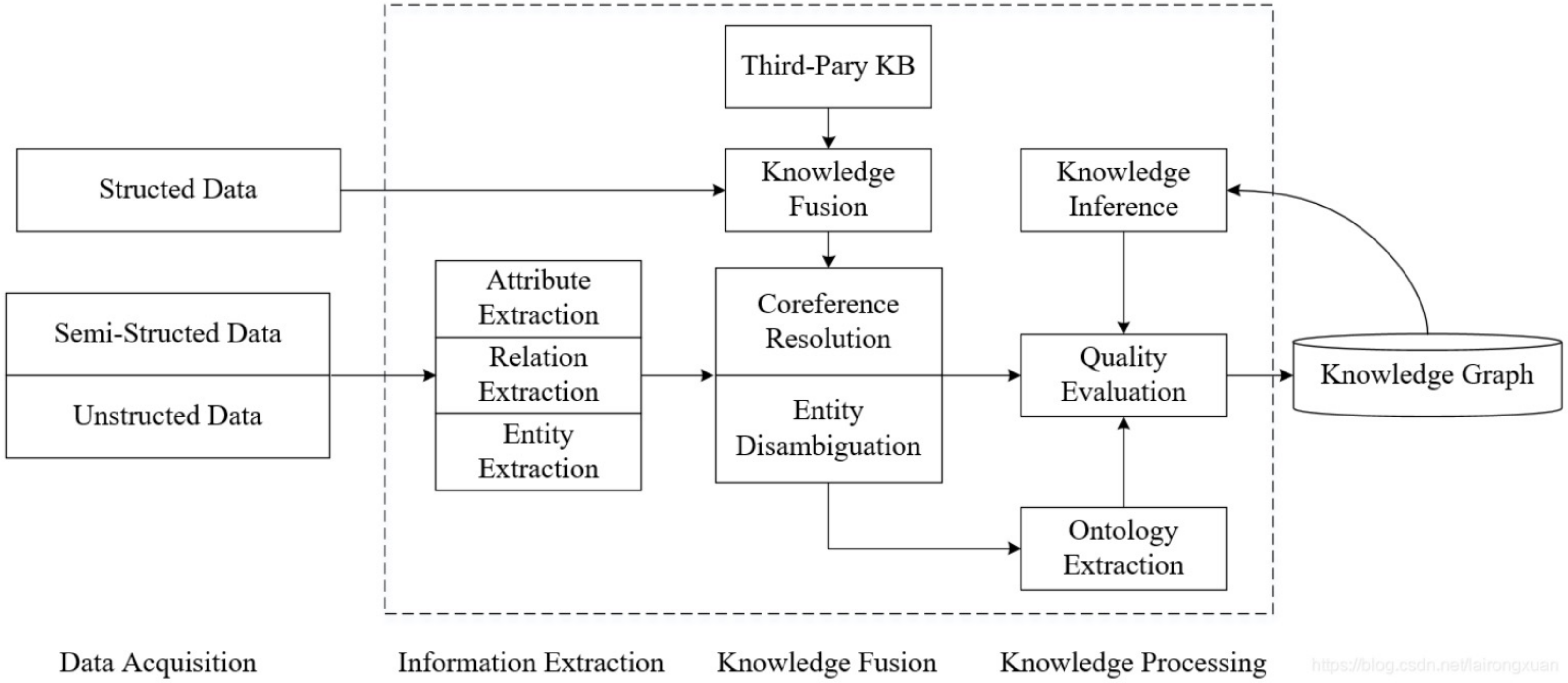
Part 3

Knowledge Graph Construction (KGC)

Knowledge Graph Construction

What is Knowledge Graph (KG) ?

The knowledge graph is a relational network obtained by connecting all kinds of information.



Knowledge Graph Construction

❑ How to construct Knowledge Graph (KG) ?

Data Acquisition :

structure data,

non-structure data,

semi-structure data

Information Extraction :

NER

RE

Knowledge Fusion :

Coreference Resolution

Entity Disambiguation

Entity Linking

Knowledge Processing :

Ontology Extraction

Knowledge Inference

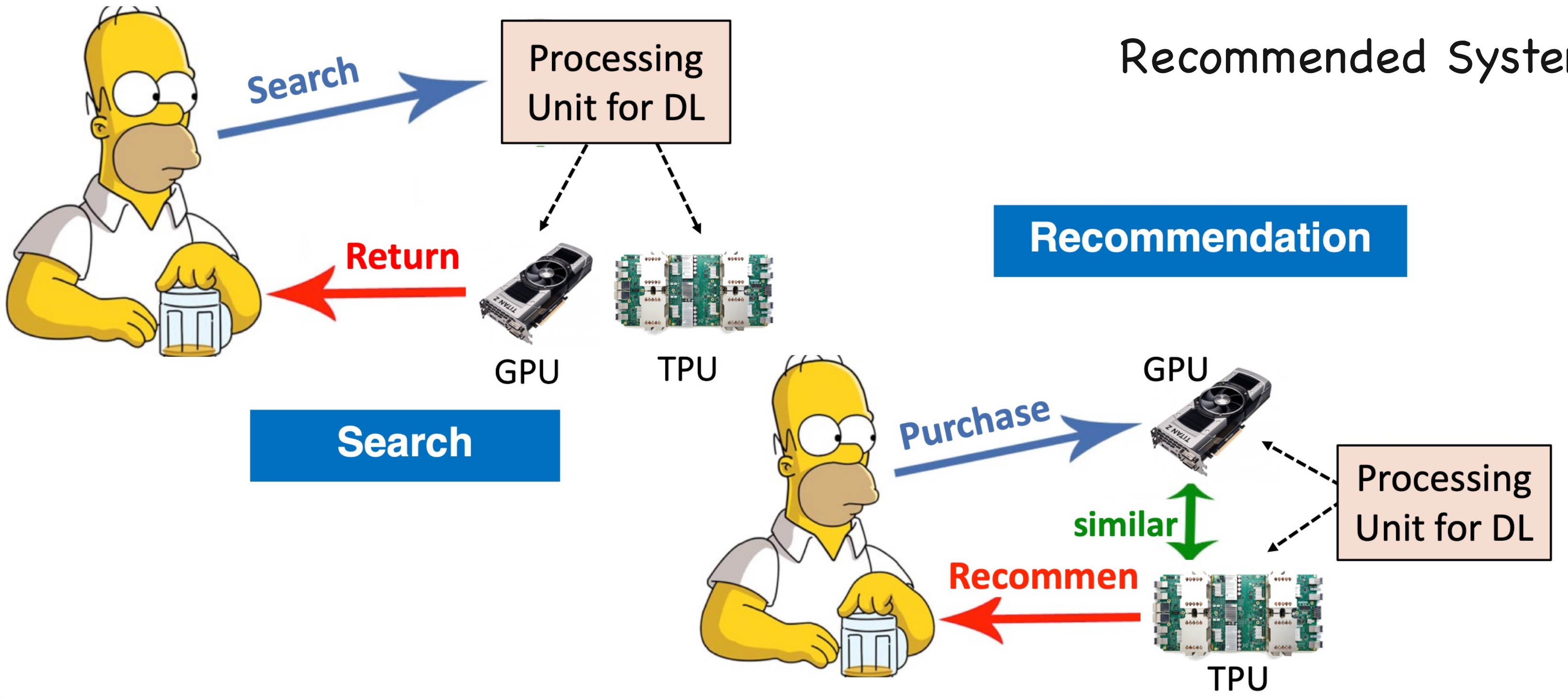
Quality Evaluation

Knowledge Graph Construction

Knowledge Graph Applications

Search Engine

Recommended System



Knowledge Graph Construction

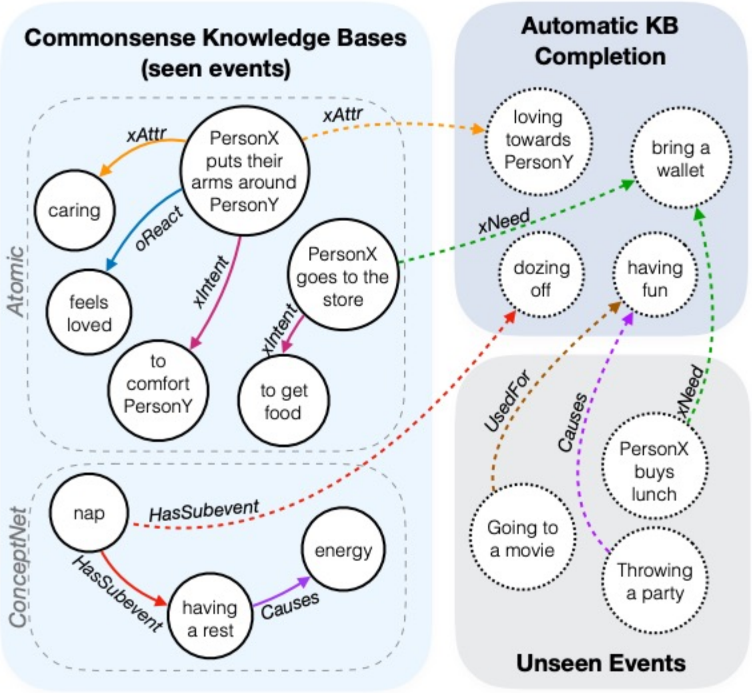
- Automatic Knowledge Graph Construction

Existing language tuple

< S R O >

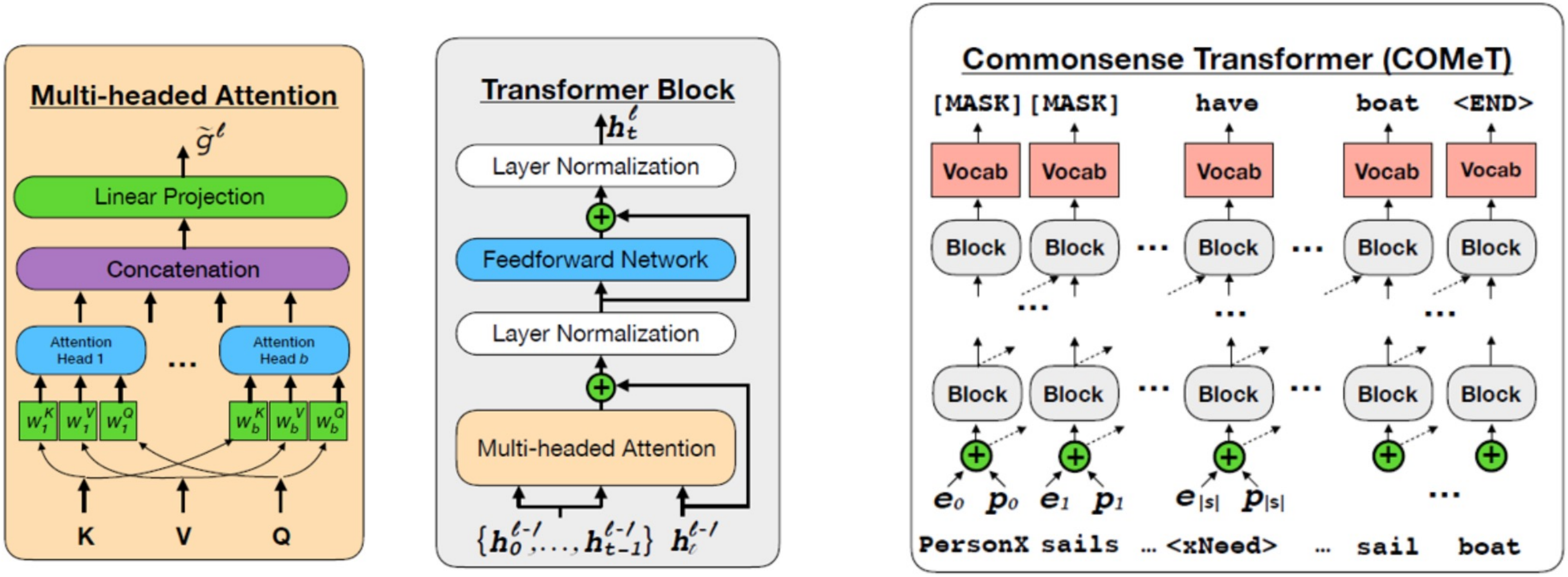
Given

Predict



Knowledge Graph Construction

- Automatic Knowledge Graph Construction



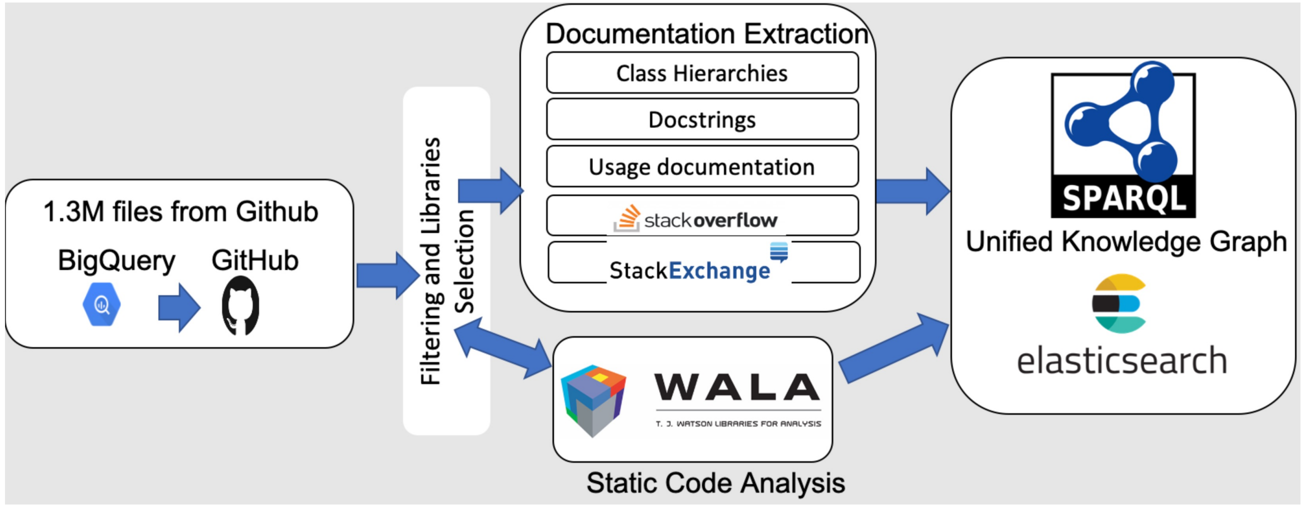
(a) (b) (c)

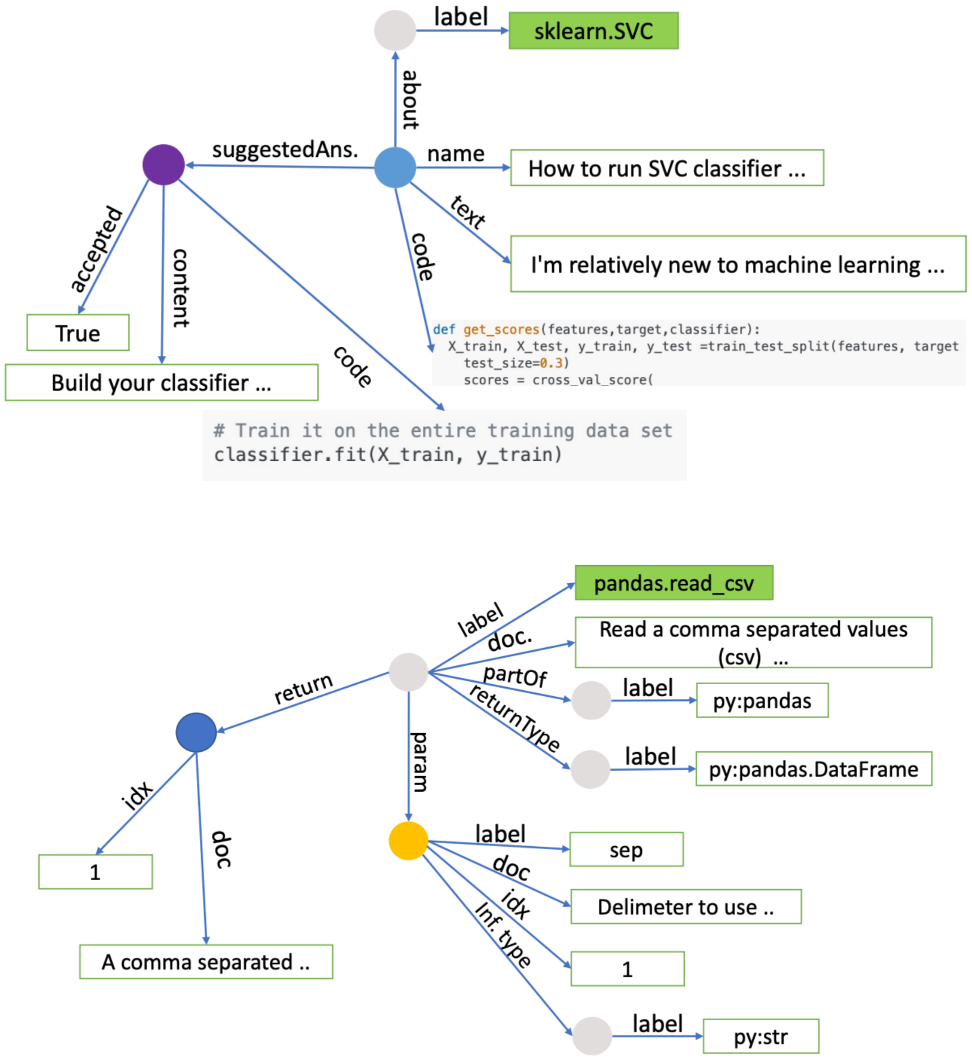
COMET: Commonsense Transformers for Automatic Knowledge Graph Construction Antoine Bosselut et.al (ACL2019)

Knowledge Graph Construction

❑ Resource Applications and Toolkits

Pipeline





A Toolkit for Generating Code Knowledge Graphs brahim Abdelaziz et.al (2020)

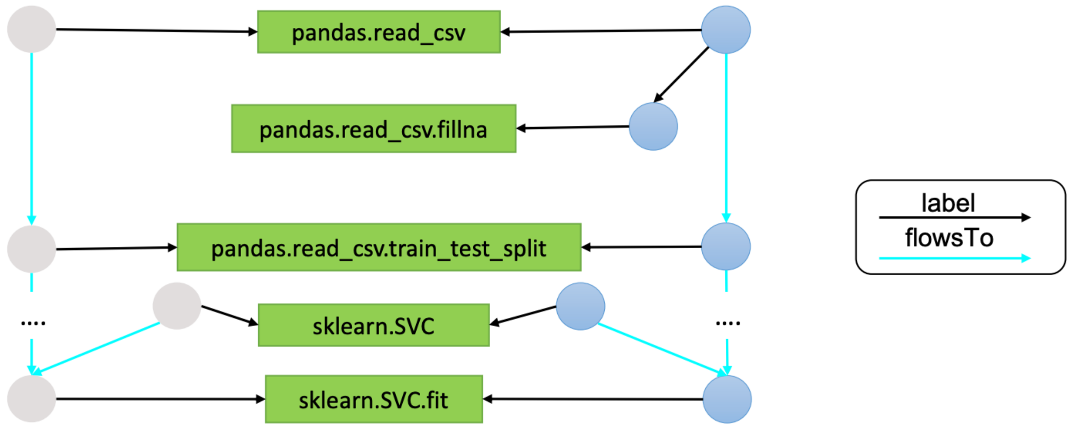
Knowledge Graph Construction

AACL IJCNLP 2022

Resource Applications and Toolkits

```
data = pd.read_csv("../input/chinese_liver_patient.csv",
    low_memory=False)

train, test  = train_test_split(X, y, test_size=0.3,
    random_state=0, stratify=data['Dataset'])

train_X = train[train.columns[:len(train.columns)-1]]
train_Y = train['Dataset']
test_X = test[test.columns[:len(test.columns)-1]]
test_Y = test['Dataset']

clf_svm_radial_basis=SVC(kernel=1.0 * RBF(1.0), random_state=0)
model = clf_svm_radial_basis.fit(train_X, train_Y)
```

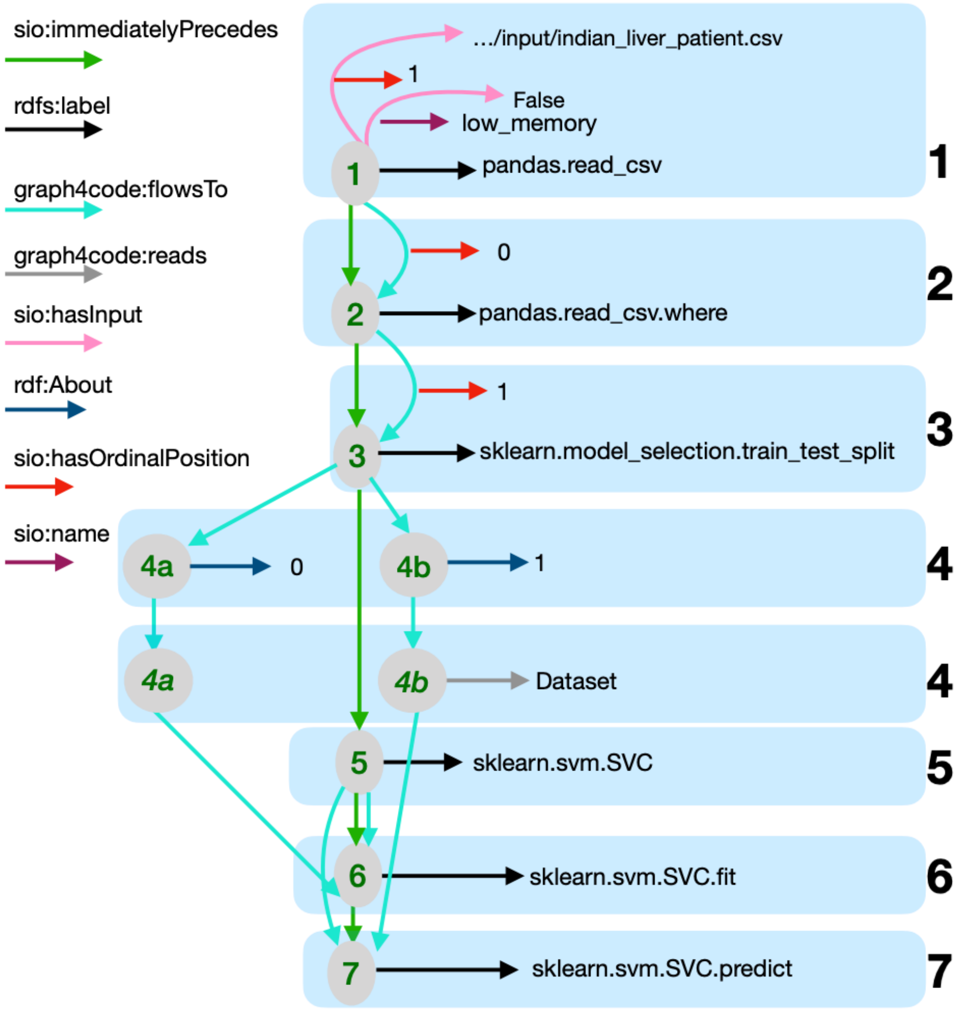
```
data = pd.read_csv("../input/indian_liver_patient.csv",
    low_memory=False)
data.fillna(method="ffill")

X = data[data.columns[:len(data.columns)-1]]
Y = data['Dataset']

train_X, test_X, train_Y, test_Y = train_test_split(X, y,
    test_size=0.3, random_state=0, stratify=data['Dataset'])

model=svm.SVC(kernel='rbf', random_state=0)
model.fit(train_X, train_Y)
```



Knowledge Graph Construction

❑ Resource Applications and Toolkits

```
data = pd.read_csv("../input/indian_liver_patient.csv", low_memory=False)   1
data = data.where(pd.notnull(data), data.median(), axis='columns')          2
train, test = train_test_split(my_df,                                       3
                               test_size = 0.3,
                               random_state = 0,
                               stratify = my_df['Dataset'])
train_X = train[train.columns[:len(train.columns)-1]]                       4a
test_X = test[test.columns[:len(test.columns)-1]]
train_Y = train['Dataset']
test_Y = test['Dataset']                                                    4b

model = svm.SVC(kernel=i, random_state=0)                                   5
model.fit(train_X,train_Y)                                                  6
prediction = model.predict(test_X)                                          7
```



Conclusion and Future work

Recent advances

- Automatic acquisition of knowledge
- Automatic fusion of multi-source knowledge
- Knowledge oriented expressive learning
- Knowledge reasoning and application

Future works

- Data acquisition
- Model performance