



# WikiNER: Brute-force Named Entity Recognition leveraging *Wikipedia* dataset

COM S 579X Natural Language Processing, Fall 2022  
December 8, 2022

---

*Mario Mastrandrea, Yuting Yang*

# What is Named Entity Recognition?

---

## Text

Iowa State University of Science and  
Technology is a public land-grant  
research university in Ames, Iowa.

## Tags

- [Organization] Iowa State University  
of Science and Technology
- [Location] Ames
- [Location] Iowa

# What is Named Entity Recognition?

## Text

[Iowa State University of Science and

B I I I I I

Technology]ORG is a public land-grant

I

research university in [Ames]LOC,

B

[Iowa]LOC.

B

## Tagging

## Tags

- [Organization] Iowa State University of Science and Technology
- [Location] Ames
- [Location] Iowa

# What is Named Entity Recognition?

## Text

[Iowa State University of Science and

B I I I I I

Technology]ORG is a public land-grant

I

research university in [Ames]LOC,

B

[Iowa]LOC.

B

## Tagging

## Tags

- [Organization] Entity 1 on *Wikipedia*
- [Location] Entity 2 on *Wikipedia*
- [Location] Entity 2 on *Wikipedia*
- .....

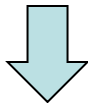
**WikiNER uses *Wikipedia* entities as Named Entities tags!**

## Why WikiNER?

---

The entities in *Wikipedia* are a good source of tags to recognize WHO, WHERE, and WHAT in a sentence

- **Massive** in the number of entities of each type (i.e., organization, locations, persons)
- Being **evolving** and expanding
- Applicable to **different domains**



Use the **top ranked *Wikipedia* Named Entities** to label a sentence with a **brute-force approach**

## How did we build **WikiNER**?

---

1. Retrieve **Wikipedia** top ranked **NEs** from *Wikidata* API
2. Build a **brute-force model** to perform NER tagging
3. **Evaluate** the model using a tagged *corpus* as a benchmark

## Retrieve *Wikipedia* top ranked NEs from *Wikidata* API

1. download **.csv file** from [Wikidata QRank project](#) with **ordered Wikipedia** entities



Entity	QRank
Q178995	219893853
Q635	113674399
Q866	93345399
...	...

2. call **Wikidata API** for each entity to retrieve its **English label**

<https://www.wikidata.org/w/api.php?action=wbgetentities&ids=Q178995&languages=en&format=json&props=labels>



3. create an output **.csv file** with the **top k Named Entities labels**

Entity	QRank	Label
Q178995	219893853	HTTP cookie
Q635	113674399	Cleopatra
Q866	93345399	YouTube
...	...	...

## Retrieve *Wikipedia* top ranked NEs - with aliases

```
def top_N_NEs(  
    input_csv_ranking_file_path: str,  
    output_csv_file_path: str,  
    top_N=1000,  
    aliases=False  
):
```



*The function we implemented to extract the labels*

- It is also possible to retrieve all the **aliases** of a Named Entity with the same API, **increasing the performances!**

<https://www.wikidata.org/w/api.php?action=wbgetentities&ids=Q178995&languages=en&format=json&props=labels|aliases>

Entity	QRank	Label
Q178995	219893853	HTTP cookie
Q178995	219893853	cookie
Q178995	219893853	web cookie
Q178995	219893853	browser cookie
Q178995	219893853	internet cookie
Q178995	219893853	cookies
Q635	113674399	Cleopatra
Q635	113674399	Cleopatra VII Philopator
Q635	113674399	Cleopatra VII
Q866	93345399	YouTube
Q866	93345399	YT
...	...	...



## Build a **brute-force model** to perform **NER tagging**

---

```
def brutal_force_NER(  
    sentence_tokens: list[str],  
    NE_list: list[str],  
    tokenizer,  
    scheme="BIO"  
):
```

*The function we implemented to tag a sentence using the Wikipedia NEs with brute-force approach*

1. Find all the **not-overlapping Matches\*** in the sentence using all the NEs (brute-force)
2. Represent this matches according to the specified **tagging scheme** (i.e., *BIO*, *BIOES*)

\*A *Match* is an **occurrence** of a Named Entity in the sentence, represented as a tuple of **indexes**  $(a,b)$ :  $a$  is the **start** index,  $b$  is the (exclusive) **ending** index

## Example of brute-force tagging

sentence\_tokens = ['I', 'went', 'to', 'school', 'at', 'Iowa', 'State', 'University', 'in', 'Fall', '2022']

NE\_list = ['Microsoft', 'Iowa State', 'Fall 2022', 'Deep Learning', 'Iowa State University', '2022', 'school']

(3,4), (5,8), (9,11)



I went to school at Iowa State University in Fall 2022  
O O O B O B I I O B I  
O O O U O B I L O B I

*Matches*

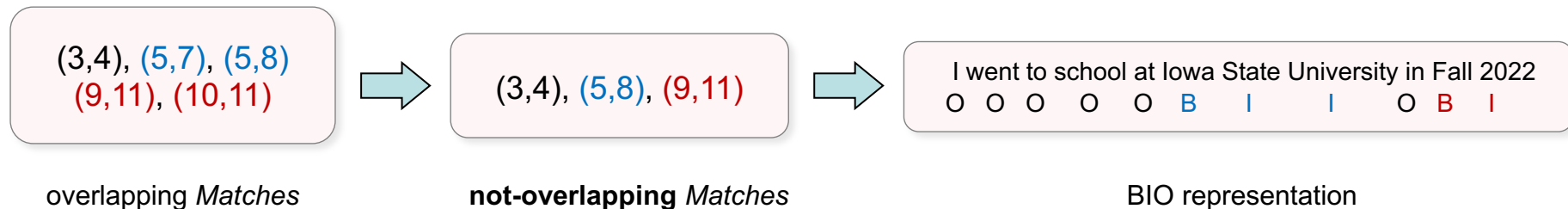
BIO and BILOU representations

## Challenge #1: Overlapping NEs

- We solved the problem of **overlapping** Named Entities prioritizing always the **longest Named Entity**

sentence\_tokens = ['I', 'went', 'to', 'school', 'at', 'Iowa', 'State', 'University', 'in', 'Fall', '2022']

NE\_list = ['Microsoft', 'Iowa State', 'Fall 2022', 'Deep Learning', 'Iowa State University', '2022', 'school']



## Supported tagging schemes

---

- Our **WikiNER** model supports 2 different tagging schemes:

**BIO:**

**B** – Beginning

**I** – Inside

**O** – Outside

**BILOU:**

**B** – Beginning

**I** – Inside

**L** – Last

**O** – Outside

**U** – Unit

## Challenge #2: Entity type (LOC, PER, ORG, MISC, etc.)

---

- NER problem also expects to find the **type** of each Named Entity (LOCation, PERson, ORGanization, etc.)
- But this is **impossible** using just a the *Wikidata* API, since it does **not** provide the type of an entity
- **Solution:** using another model (like **BERT**) to also find the type of NEs (*to be continued*)

## Evaluate the model using a tagged *corpus* as benchmark

- Use **CoNLL2003 dataset** as benchmark
  - It contains a *corpus* of sentences whose Named Entities have been tagged in their respective **types** using the **BIO scheme**
- Use the **WikiNER** model to tag its sentences
- Evaluate the **performances** comparing the **predicted tags** with the **real tags** in the *corpus* (ignoring the NE type)

token	POS tag	chunk tag	NER tag
EU	NNP	B-NP	B-ORG
rejects	VBZ	B-VP	O
German	JJ	B-NP	B-MISC
call	NN	I-NP	O
to	TO	B-VP	O
boycott	VB	I-VP	O
British	JJ	B-NP	B-MISC
lamb	NN	I-NP	O
.	.	O	O
Peter	NNP	B-NP	B-PER
...	...	...	...

## Evaluation metrics

- **Precision**

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Among all the predicted NEs, how many of them were **true NEs**

- **Recall**

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Among all the real NEs, how many of them we were able to **predict**

- **F1-score**

$$2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

**Harmonic mean** between *precision* and *recall*

## WikiNER performances on CoNLL2003 dataset

---

- We used **seqeval** library to compute *precision*, *recall* and *F1-score*
- **Compare** the results using **different numbers** of Wikipedia top NEs

#NEs	Precision	Recall	F1-score
100	0.94	0.05	0.09
1,000	0.89	0.14	0.24
10,000	0.26	0.22	0.24

*with **no** aliases*

#NEs	Precision	Recall	F1-score
100	0.82	0.06	0.11
1,000	0.44	0.17	<b>0.25</b>
10,000	0.15	<b>0.29</b>	0.20

*with aliases*



## Timing problems

---

- *WikiNER* is a very **time-consuming** approach!
  - **extracting** *Wikipedia* NEs
  - **executing** the brute-force model

#NEs	time
100	~20s
1,000	~3min
10,000	~30min

estimated times to **extract** NEs

#NEs	no aliases	with aliases
100	~5min	~20min
1,000	~30min	~2h
10,000	~3h	~12h

estimated times to execute **WikiNER** on **CoNLL2003**

## Conclusions

---

- *WikiNER* brutal force model is **ineffective** to perform a NER task
  - **Low performances** (max recall: 29% - 10,000 NEs and aliases)
  - **Time-consuming** (it takes hours)
  - **No NEs types**
- Two issues in *Wikipedia* entities:
  - The presence of **more NEs** naturally **decreases precision** (although **increases recall**)
  - The *Wikipedia* entities are **not universal** enough to capture benchmark entities
- Better to resort to more sophisticated models such as **BERT** (next step)

Questions?

**Thank You!**