McGill at CRAC 2023:
Multilingual Generalization of Entity-Ranking Coreference Resolution Models

Ian Porada & Jackie Chi Kit Cheung

Mila, McGill University
ian.porada@mail.mcgill.ca

Overview

We apply the **entity-ranking model** originally proposed by **Xia et al. (2020)**.*

I'll go over:

1. Model details
2. Things we tried

*Incremental Neural Coreference Resolution in Constant Memory (Xia et al., EMNLP 2020)

Model

Intuition

Process spans left to right:
score each *candidate mention* pairwise against a running list of entities.

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.

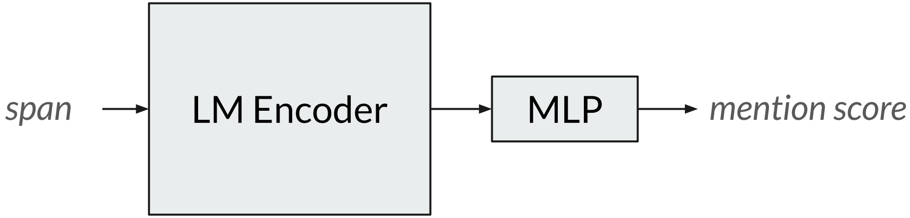
- Calculate a set of all spans <= 20 tokens
    - E.g. {`"he", "ate", "an", …, "he ate", "ate an", …, "he ate an", …`}

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.

- Calculate a set of all spans <= 20 tokens
  - E.g. {`“he”, “ate”, “an”, …, “he ate”, “ate an”, …, “he ate an”, …`}
- Score each span using a task-specific head

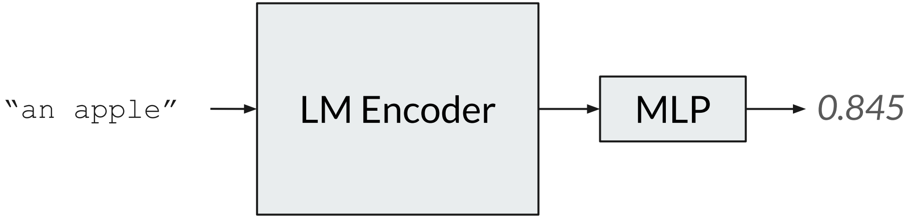


Example

Input document: "`He ate an apple. It tasted good.`"

Step 1: Find mention candidates.

- Calculate a set of all spans <= 20 tokens
    - E.g. {`"he", "ate", "an", …, "he ate", "ate an", …, "he ate an", …`}
- Score each span using a task-specific head



Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.

- Calculate a set of all spans <= 20 tokens
    - E.g. {`“he”, “ate”, “an”, …, “he ate”, “ate an”, …, “he ate an”, …`}
- Score each span using a task-specific head
- Keep the spans with the top “**0.4 * n**” mention scores as candidate mentions
    - E.g. {`“he”, “ate”, “an apple”, “it”, “good”, “he ate an”, …`}

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.

- Calculate a set of all spans <= 20 tokens
  - E.g. {`“he”, “ate”, “an”, …, “he ate”, “ate an”, …, “he ate an”, …`}
- Score each span using a task-specific head
- Keep the spans with the top “**0.4 * n**” mention scores as candidate mentions
  - E.g. {`“he”, “ate”, “an apple”, “it”, “good”, “he ate an”, …`}
- Remove all spans with a negative score
  - E.g. candidate mentions = {`“he”, “ate”, “an apple”, “it”, “good”`}

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

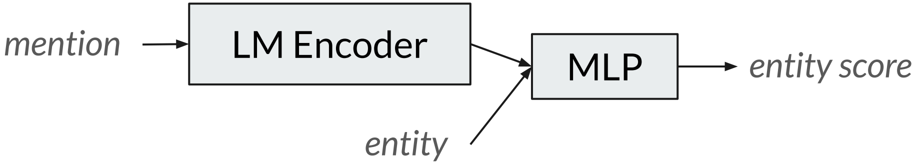
- Initialize a set of entities E = {}
- Process mentions left to right

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head



Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head
    - If all entities scores are < 0, add a new entity to E
    - Otherwise, the mention refers to the entity with the highest score

Example

Input document: “`He ate an apple. It tasted good.`”

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head
    - If all entities scores are < 0, add a new entity to E
    - Otherwise, the mention refers to the entity with the highest score
      - Update the representation of that entity

Example

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head
    - If all entities scores are < 0, add a new entity to E
    - Otherwise, the mention refers to the entity with the highest score
      - Update the representation of that entity

Input document: "`He ate an apple. It tasted good.`"

mentions = {`"he", "an apple", "it"`}

1. E = {*he*}
2. E=

Example

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head
    - If all entities scores are < 0, add a new entity to E
    - Otherwise, the mention refers to the entity with the highest score
      - Update the representation of that entity

Input document: “`He ate an apple. It tasted good.`”

mentions = {`“he”, “an apple”, “it”`}

1. E = {*he*}
2. E = {*he, an apple*}

Example

Step 1: Find mention candidates.
Step 2: Assign each mention candidate to an entity (i.e., a cluster).

- Initialize a set of entities E = {}
- Process mentions left to right
  - Score the current mention against all entities in E using a task-specific head
    - If all entities scores are < 0, add a new entity to E
    - Otherwise, the mention refers to the entity with the highest score
      - Update the representation of that entity

Input document: “`He ate an apple. It tasted good.`”

mentions = {`“he”, “an apple”, “it”`}

1. E = {*he*}
2. E = {*he, an apple*}
3. E = {*he, (an apple, it)*}

Experiments

Data Processing

1. **Input**: Use **udapi** to convert CorefUD to json
    a. For GUM, we extract speaker information from headers
    b. We treat zero anaphors as the underscore token ("_")
2. **Output**: Use `udapy -s corefud.MoveHead` to calculate syntactic heads

Experiment 2: Mixing strategy

Mixing strategy (XLM-R base encoder)	**CoNLL F1 (exact match)**
Uniform	62.78
Proportional	64.68
Proportional w.r.t. = min(size, 500)	**64.86**

Experiment 1: Encoder

Encoder (large size)	CoNLL F1 (exact match)
XLM-Roberta	**67.32**
MT5	64.76

Conclusion

Conclusion

Entity-ranking models serve as **a reasonable baseline** with some simple adaptations.

system	head-match	partial-match	exact-match	with singletons
1. CorPipe	74.90	73.33	71.46	76.82
2. Anonymous	70.41	69.23	67.09	73.20
3. Ondfa	69.19	68.93	53.01	68.37
4. **McGill**	65.43	64.56	63.13	68.23
5. DeepBlueAI	62.29	61.32	59.95	54.51
6. DFKI-Adapt	61.86	60.83	59.18	53.94
7. ITUNLP	59.53	58.49	56.89	52.07
8. BASELINE	56.96	56.28	54.75	49.32
9. DFKI-MPrompt	53.76	51.62	50.42	46.83