Entity and Relation Extraction

Department of Information Engineering and Mathematical Sciences
University of Siena - Siena, Italy

Andrea Zugarini

SAILab

December 5th, 2019

Information Extraction

Text is a huge source of information!

Information Extraction (IE) is one of the most important topics in NLP, and it is about extracting **structured** information from **unstructured** text (documents).

Goal: align textual spans to a **Knowledge Base** KB.

KBs typically store **factual** data into a **triple**-based ontology.

Knowledge Base
Triple-based

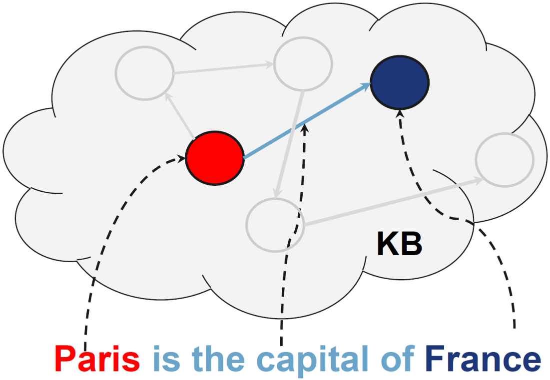
In its simplest version, a KB stores a fact as a triple of two **entities** and a **relation**: (e_i, r_k, e_j).

Often, both entities and relations may belong to a **type** of the ontology.

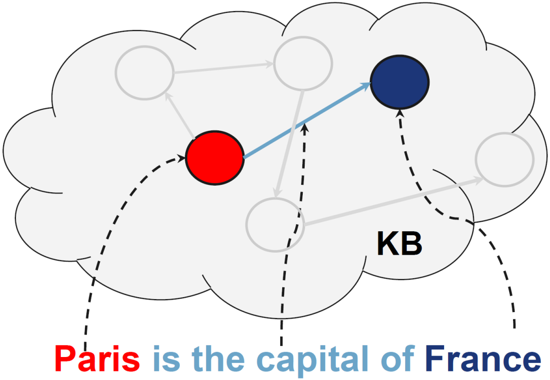
A structured representation of the information it is easier to handle automatically than plain text, allowing efficient storing and retrieval.

A knowledge base can be represented with a directed **graph**, where entities are **nodes** and relations are **edges**.

Information Extraction has to align textual spans of entities and relation to their respective nodes and edges in the KB.



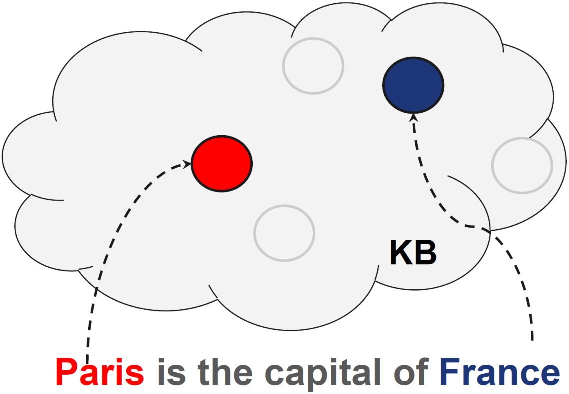
Textual spans are referred as **mentions**.



An entity is a unique instance of something in the real world.

The same entity may be referred by multiple mentions, coreferences included.

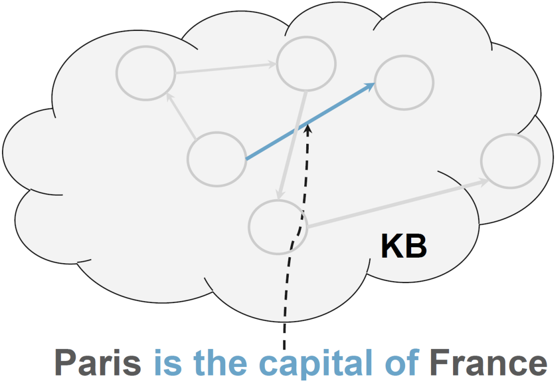
Problem related to **Named Entity Recognition**.



A relation is a property that connects two (or more) entities.

Challenging problem: there are **many** ways to express the same relations.

Very similar to **Relation Extraction**.



Can plain deep learning techniques be applied for Entity linking and Relation Extraction? Yes, however:

- End-to-end approaches (encoder-decoder) are very good at mapping text into new text, but they **do not build any concrete understanding model**
- What does **"understanding text"** mean?

UNDERSTANDING: *mapping text onto a structured (factual) knowledge base* (**Entity Linking**)

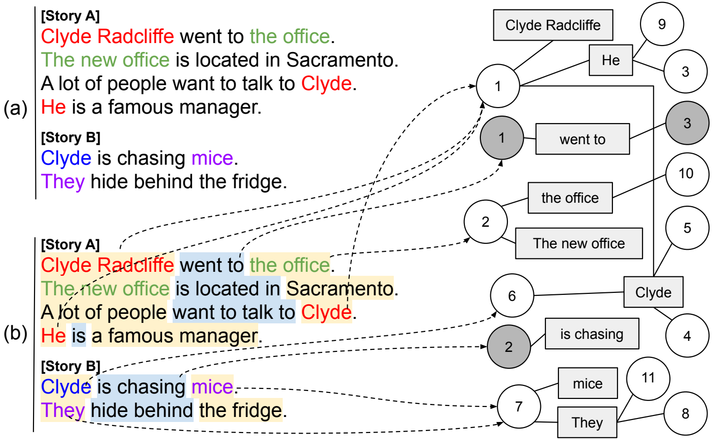
TEXT STREAMS: *we have to incrementally build and update the knowledge while we process a text stream!*

Problem Setting

- We consider a continuous stream of text
- Groups of sentences organized into small stories about a (not-known-in-advance) set of actors/objects - m mentions and n entities/relations
- The narration is discontinuous whenever a new story begins

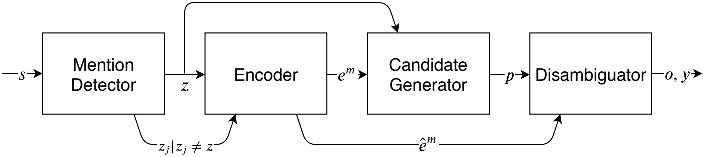
Challenges
KB construction, Online Learning, Entity Discovery, Entity Linking, Multiple Stories

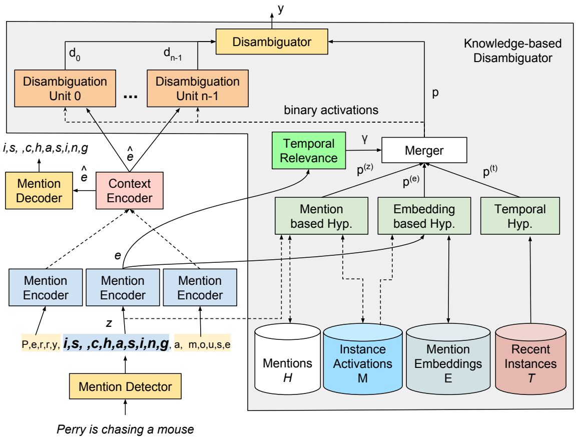
Problem Setting
An Example



The system is the composition of multiple modules.

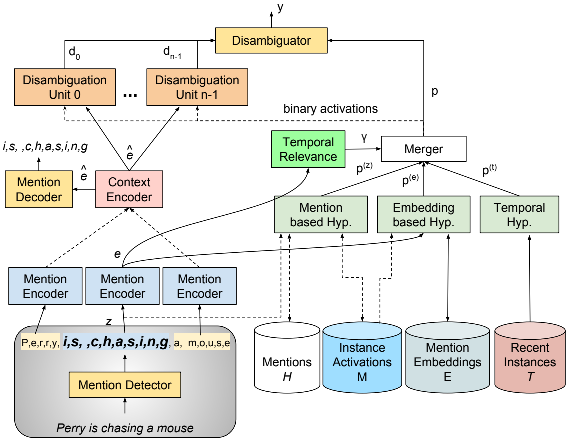
- **Mention Detector**: Segment each sentence in non-overlapping text fragments.
- **Encoder**: textual mention and its context are encoded into a vectorial representation.
- **Candidate Generator**: given an input mention z, the candidate generator implements memory components that are used to generate a list of compatible candidates from the KB.
- **Disambiguator**: Based on the mention context, the disambiguator is responsible of determining which candidate is the most likely.





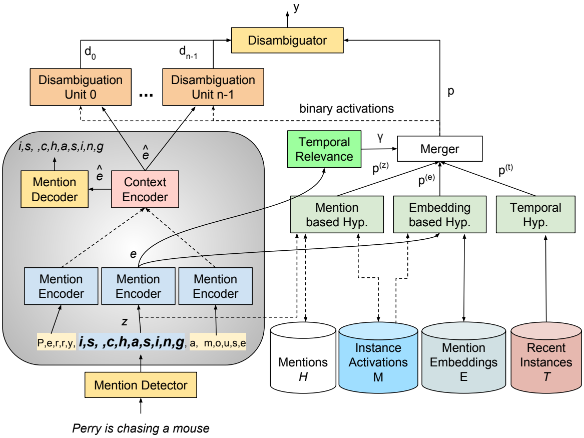
y
Disambiguator
Knowledge-based Disambiguator
d0
dn-1
Disambiguation Unit 0
...
Disambiguation Unit n-1
p
binary activations
Temporal Relevance
γ
Merger
i,s, ,c,h,a,s,i,n,g
ê
Mention Decoder
ê
Context Encoder
p(z)
p(e)
p(t)
Mention based Hyp.
Embedding based Hyp.
Temporal Hyp.
e
Mention Encoder
Mention Encoder
Mention Encoder
z
P,e,r,r,y, i,s, ,c,h,a,s,i,n,g, a, m,o,u,s,e
Mentions H
Instance Activations M
Mention Embeddings E
Recent Instances T
Mention Detector
Perry is chasing a mouse

Architecture
Segmentation



Focus the attention only on relevant text spans (mentions to entities or relations). **How**?

1. Supervised learning using syntax-based generated labels
2. Pre-trained **character-based** model to spot both entities and relations
3. Post processing of predictions to adjust misplaced markers (unclosed elements etc...)



y
Disambiguator
d_0
d_{n-1}
Disambiguation Unit 0
...
Disambiguation Unit n-1
p
binary activations
i,s, ,c,h,a,s,i,n,g
ê
ê
Mention Decoder
Context Encoder
Temporal Relevance
γ
Merger
$p^{(z)}$
$p^{(e)}$
$p^{(t)}$
Mention based Hyp.
Embedding based Hyp.
Temporal Hyp.
e
Mention Encoder
Mention Encoder
Mention Encoder
z
P,e,r,r,y, i,s, ,c,h,a,s,i,n,g, a, m,o,u,s,e
Mentions H
Instance Activations M
Mention Embeddings E
Recent Instances T
Mention Detector
Perry is chasing a mouse

We are given a sequence of *segments*, that are mentions to entities or relations

- **Represent segments**: each segment z_i is processed as a *sequence of characters* and it is embedded into e_i

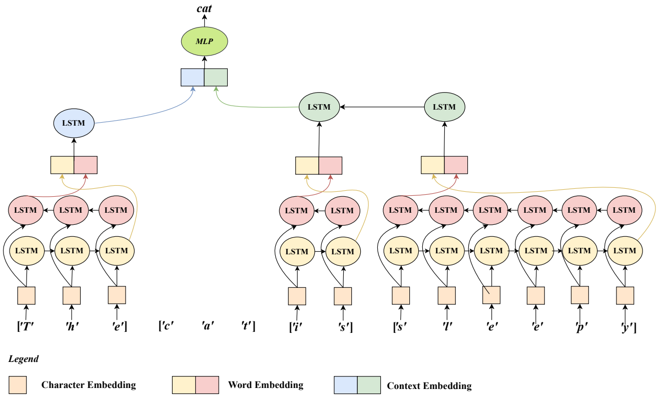
$$e_i = enc(z_i) \quad = \quad \left[\overrightarrow{eRNN}(c_{i,1}, \ldots, c_{i,|z_i|}),\ \overleftarrow{eRNN}(c_{i,1}, \ldots, c_{i,|z_i|})\right]$$

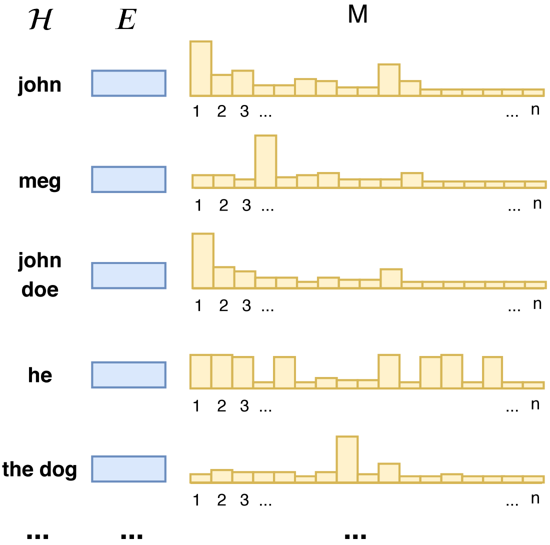
- **Represent contexts**: the context around z_i is embedded into the representation $\hat{e}_i$ (that does not include z_i)

$$\hat{e}_i = enc(z_i|s - z_i) \quad = \quad \left[\overrightarrow{\hat{e}RNN}(e_1, \ldots, e_{i-1}),\ \overleftarrow{\hat{e}RNN}(e_{i+1}, \ldots, e_n)\right]$$

Unsupervised Learning in a encoding-decoding scheme as (CBOW)

Sketch of mention and context encoding architecture while processing the sentence "*The cat is sleepy*" with target word *cat*





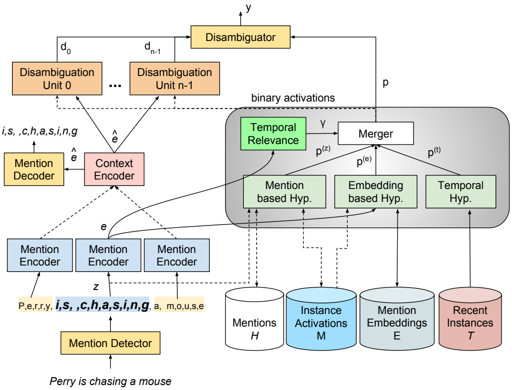
$\mathcal{H}$
E
M
john
meg
john doe
he
the dog
1 2 3 ... ... n
...

Our **Knowledge Base** is organized in 4 memory components

1. $\mathcal{H}$ is the set of mentions (raw text)
2. E is the matrix of the embeddings of each mention
3. $\mathcal{T}$ buffers the last disambiguated instances, resets at the beginning of a story
4. M is a matrix where is row is associated to a mention $z \in \mathcal{H}$ and $\sigma(M_{\mathcal{H}(z)})$ provides the activation scores of currently known instances

Anytime a new element is encountered the **KB** is updated

Architecture
Hypoteses Formulation



Given a mention z and its embedding e at time t, three hypotheses are formulated

- **Mention-based**: $\boldsymbol{p}^{(z)} = \sigma(M_{\mathcal{H}(z)})$
- **Embedding-based**:

$$\boldsymbol{p}^{(e)} = \left(\left[\frac{\cos(e, E_i) + 1}{\sum_{j=1}^{m} \cos(e, E_j) + m} \right]_{i=1}^{m} \right)' \cdot \sigma(M)$$

- **Time-based**:

$$\boldsymbol{p}^{(t)} = \frac{[u(i, \mathcal{T})]_{i=1}^{n}}{\max\left[u(j, \mathcal{T})\right]_{j=1}^{n}}$$

Ensembler

$$\boldsymbol{p} = (1-\gamma) \cdot \left(\boldsymbol{p}^{(z)} + (1 - \boldsymbol{p}^{(z)})\boldsymbol{p}^{(e)} \right) + \gamma \cdot \boldsymbol{p}^{(t)}$$

Given a mention z and its embedding e at time t, three hypotheses are formulated

- **Mention-based**: $\boldsymbol{p}^{(z)} = \sigma(M_{\mathcal{H}(z)})$
- **Embedding-based**:

$$\boldsymbol{p}^{(e)} = \left(\left[\frac{\cos(e, E_i) + 1}{\sum_{j=1}^{m} \cos(e, E_j) + m} \right]_{i=1}^{m} \right)' \cdot \sigma(M)$$

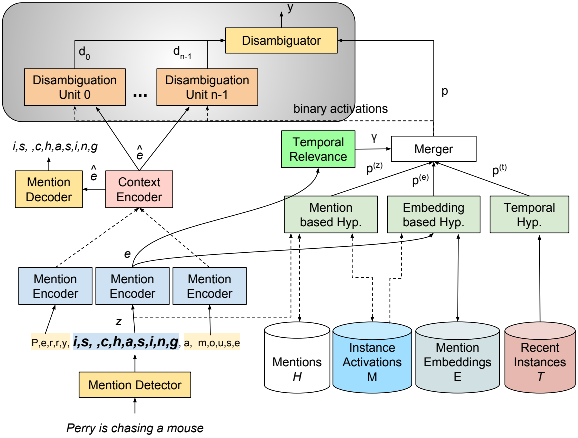
- **Time-based**:

$$\boldsymbol{p}^{(t)} = \frac{[u(i, \mathcal{T})]_{i=1}^{n}}{\max \left[u(j, \mathcal{T})\right]_{j=1}^{n}}$$

Ensembler

$$\boldsymbol{p} = (1 - \gamma) \cdot \left(\boldsymbol{p}^{(z)} + (1 - \boldsymbol{p}^{(z)}) \boldsymbol{p}^{(e)} \right) + \gamma \cdot \boldsymbol{p}^{(t)}$$

Architecture
Disambiguator



Hypotheses outputs potential candidates, **disambiguation** resolves ambiguities by selecting the correct one(s). **How**?

It looks at the **context** to find most compatible mention wrt n disambiguation units

- **Disambiguation Unit**: given the context $\hat{e}$, predicts the activation of the instance (Predictors have a local support around κ centres)

$$d_i(\hat{e}) = \frac{1}{2} + \frac{1}{2} \max_{j=1}^{\kappa} \cos(\hat{e}, \hat{w}_{ij})$$

Final output of the system is $\boldsymbol{o}$

$$\boldsymbol{o} = \delta(\boldsymbol{p} > \tau_r) \cdot (\eta \cdot \boldsymbol{p} + (1 - \eta) \cdot \boldsymbol{d})$$

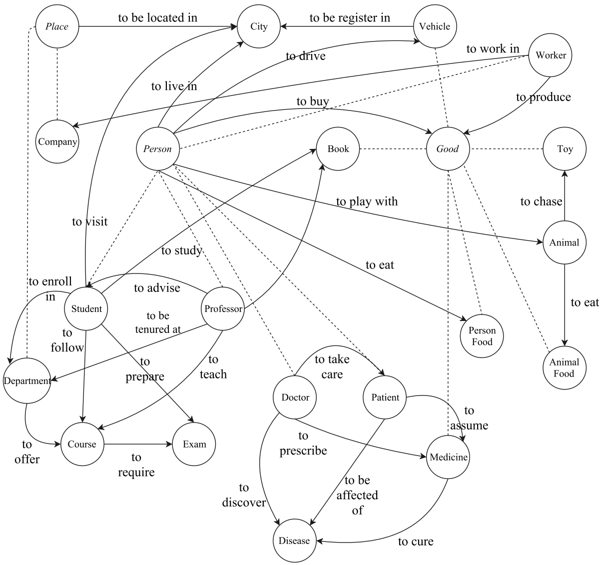
Online Learning process accordingly to either supervision or **self learning**

When no supervision is provided, we distinguish among three cases:

i. $\max \boldsymbol{o} \geq \tau_a$: RECOGNIZED SOME INSTANCES
ii. $\max \boldsymbol{p} > \tau_r \wedge \max \boldsymbol{o} < \tau_a$: UNCERTAINTY
iii. $\max \boldsymbol{p} \leq \tau_r$: UNKNOWN INSTANCE

Learnable parameters: M, disambiguation units $\boldsymbol{d}$ and γ

- Collection of $10k$ sentences organized in 564 stories
- A story is a list of not repeated facts mostly focussed on a certain entity, also called **main entity**
- 130 **entity** and 27 **relation** instances, belonging to a pre-designed ontology
- Overall there are 2176 single word tokens, 1528 and 288 mentions to entities and relations, about 6830 ambiguous mentions.



Place
to be located in
City
to be register in
Vehicle
to work in
Worker
to drive
to live in
to produce
to buy
Company
Person
Book
Good
Toy
to play with
to chase
to visit
Animal
to study
to eat
to enroll in
to advise
to eat
Student
Professor
Person Food
to be tenured at
to follow
to prepare
to teach
to take care
Animal Food
Department
Doctor
Patient
to assume
to offer
Course
Exam
to require
to prescribe
Medicine
to discover
to be affected of
Disease
to cure

- Wikipedia pages are loosely aligned with Freebase triples
- Composed of a collection of summaries, each being a description of a certain entity. Each summary is considered as a story.
- About $560k$ **entities** extracted from $10k$ pages, we took a sub-portion of 1112 pages.
- We marked text between two entities as relation.

Evaluation Setting

Each story split into two parts: a supervised and an unsupervised one.

Accuracy on each prediction is measured at the same time when the prediction is made.

Two results reported:

- All the unsupervised sentences of a story (**ALL**)
- Only the last sentence of the story (**LAST**)

Competitors

Rule-Based

An informed model that buffer statistics on the supervisions received up to time t.

Already seen mention: predicts the most common supervision

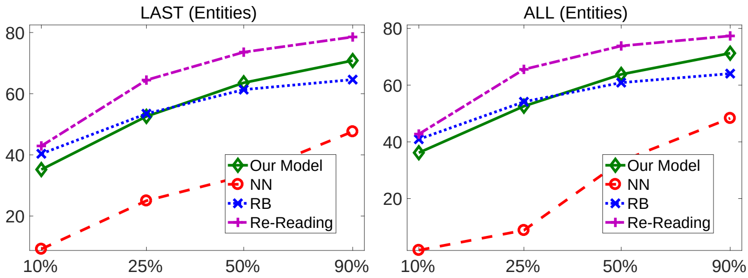
Never seen mention: responds with the most frequent supervision of the story

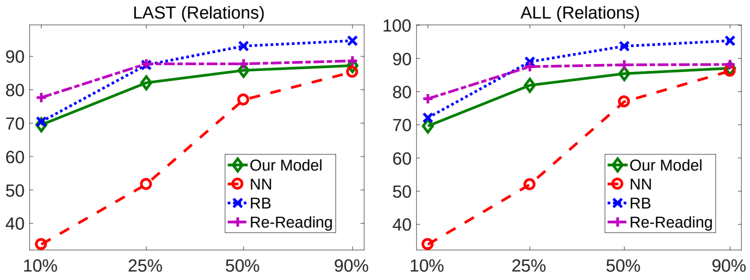
Deep-RNN

A simple neural mention classifier

- Input $[e, \hat{e}]$
- 1 hidden layer of size 600
- Softmax activation in the output layer

NB: Both models always predicts on ground truth mentions!


LAST (Entities)
80
60
40
20
10%
25%
50%
90%
Our Model
NN
RB
Re-Reading
ALL (Entities)
80
60
40
20
10%
25%
50%
90%
Our Model
NN
RB
Re-Reading


LAST (Relations)
90
80
70
60
50
40
10%
25%
50%
90%
Our Model
NN
RB
Re-Reading
ALL (Relations)
100
90
80
70
60
50
40
10%
25%
50%
90%
Our Model
NN
RB
Re-Reading

	Model	10%	25%	50%	90%
All	RB	16.84	40.44	48.28	49.55
	Deep-RNN	0.6	3.01	12.34	21.78
	Our Model	39.25	**54.57**	**69.64**	**75.45**
	Re-Reading	**44.75**	**54.66**	66.88	70.55
Last	RB	17.28	40.87	48.04	49.37
	Deep-RNN	0.6	3.25	12.11	21.37
	Our Model	37.44	52.93	**67.45**	**75.37**
	Re-Reading	**43.41**	**53.38**	65.13	70.39

Conclusions and Future Works

We presented an end-to-end model for entity/relation mentions discovery and disambiguation in text streams by constantly updating an interpretable KB

Next Steps

- Entity and Relation Types introduction
- Higher-level reasoning
- Dynamic KB re-organization (pruning, merging etc..)

Thank You !!!