Topic-Based Annotations

Up to now we always used a term-based vector space model, where we use the cosine between a query and a document to measure the similarity. However this approach has some issues if we and synonymy and polysemy in a document or in a query.

EXAMPLE

"MICROSOFT'S BROWSER"

SYNONYMY

"INTERNET EXPLORER"

DIFFERENT TERMS, SAME MEANINGS

"PAPARAZZI PHOTOGRAPHING STARS"

POLYSEMY

"ASTRONOMER PHOTOGRAPHING STARS"

SIMILAR PHRASES, DIFFERENT MEANINGS

We would like to be able to retrieve the information needed by the user.

We do this using a knowledge graph, an hand-crafted graph where the nodes represent entities and the edges represent relation.

A knowledge graph contains context-aware informations and the idea of the topic-based annotation is to find the anchors in the text and annotate them with the correct entities drawn from this graph.

THE MOST KNOWN KNOWLEDGE GRAPH IS WIKIPEDIA

EXAMPLE

"MICROSOFT'S BROWSER"

"INTERNET EXPLORER"

IE WIKIPEDIA PAGE

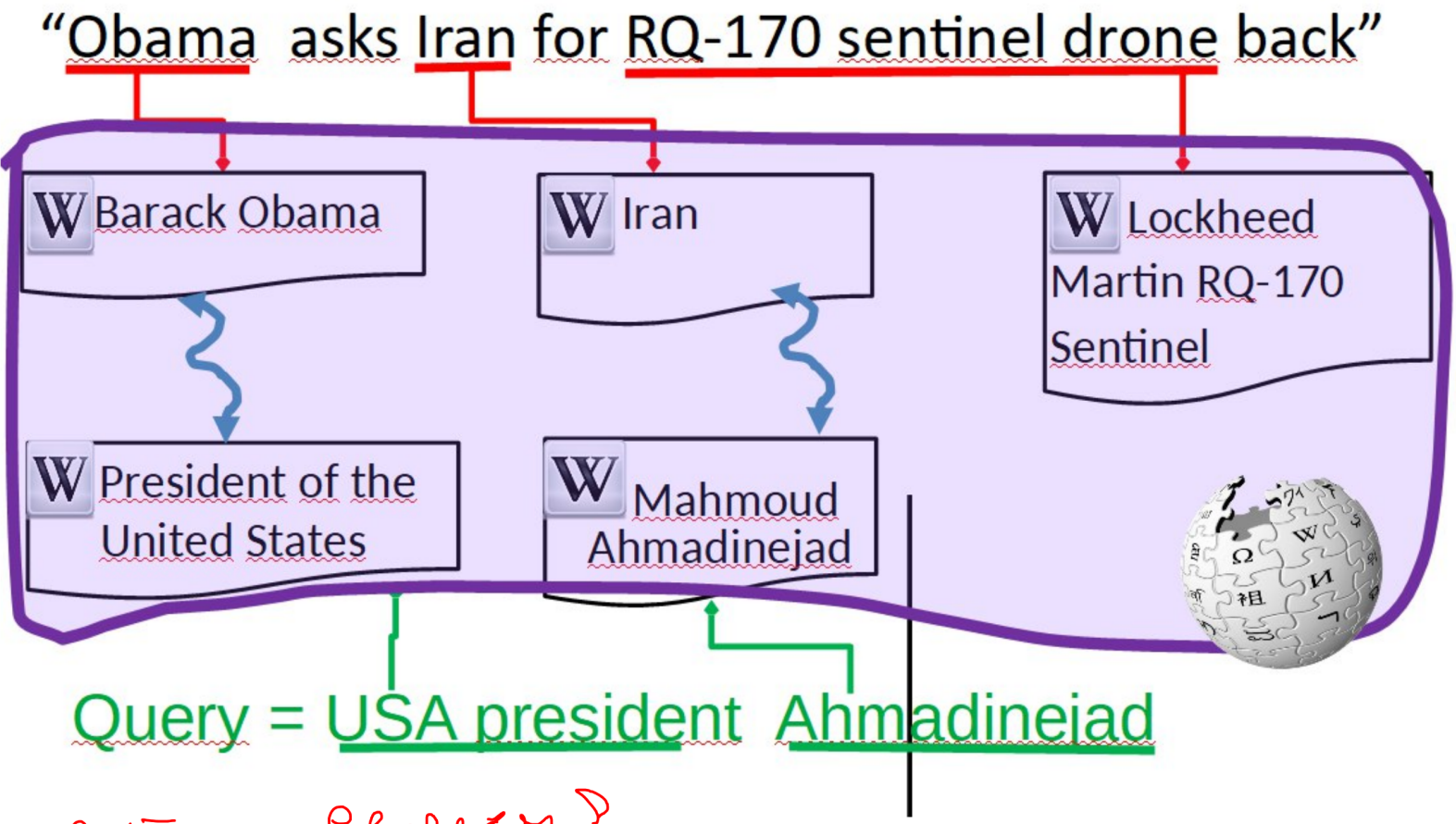
"PAPARAZZI PHOTOGRAPHING STARS"

CELEBRITY WIKIPEDIA PAGE

"ASTRONOMER PHOTOGRAPHING STARS"

STAR WIKIPEDIA PAGE

EXAMPLE OF QUERY



BUT PROBLEM !!

"DIEGO MARADONA WON AGAINST MEXICO"

DIEGO MARADONA — ANCHOR A

MEXICO — ANCHOR B

CANDIDATE PAGES OF A:
- Ex-Argentina's Coach
- DIEGO MARADONA (the nephew)
- MARADONA STADIUM

CANDIDATE PAGES OF B:
- MEXICO NATION
- MEXICO FOOTBALL
- MEXICO BASEBALL

How do we pick the correct entities (WIKI PAGES) for our TWO Anchors? Clearly they are related and the key is to quantify the relatedness between their candidate pages.

So how do we map our two anchors to the correct WIKIPEDIA PAGES?

NOTE: WE ARE ASSUMING THAT TWO ANCHORS ARE ENOUGH TO DISAMBIGUATE THE CONTEXT (for simplicity)

Here what we need to link an anchor a with the correct page p:

- **Context of a**

  Given the text T, the context of a are all the words around the anchor

  $$T = \ldots w_1 w_2 w_3 \; a \; w_4 w_5 w_6 \ldots$$

- **Content of p**

  Simply the content $p = z_1 z_2 z_3 \ldots$

- **Link probability measure**

  Measures the "goodness" of an anchor a (in how many Wiki pages it appears as a link?)

  $$Lp(a) = \frac{\text{freq. of } a \text{ as anchor}}{\text{freq. of } a \text{ in text}}$$

- **Commonness measure**

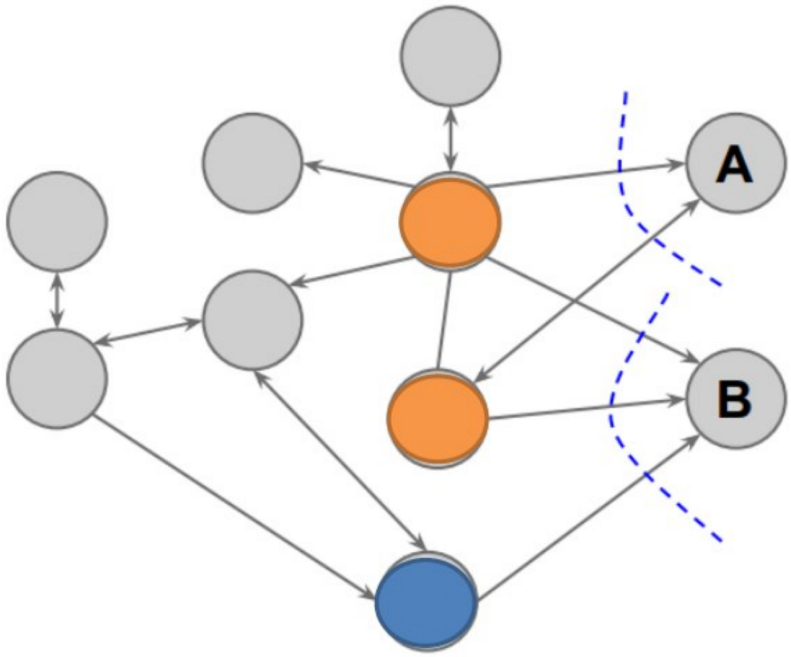
Measures how many times a certain anchor a links to the same page p

$$P(p|a) = \frac{\#\, a \text{ linked to } p}{\#\, a \text{ as anchor}}$$

- **Relatedness between pages**

In the knowledge graph, p is a entity node and a is a mention node

$$sim(A,B) = \frac{\log(\max(|A|,|B|)) - \log(|A \cap B|)}{\log(|W|) - \log(\min(|A|,|B|))}$$



Let's see now a concrete example of an annotator, TagMe, and see how it checks for feasible anchors

TagMe

A tool to identify meaningful short-phrases in a free text and link them to a Wikipedia page.

(= It analyses the text, obtain a set of anchor and associate them with Wikipedia pages

BY USING THE FEATURES DESCRIBED ABOVE, TAGME IS ABLE TO DISAMBIGUATE BETWEEN THE PAGES THROUGH A VOTING SCHEME

TagMe steps

1) Given a text, parse it and detect the texts to annotate (anchors)

2) Retrive the candidate pages for each anchor

3) Disambiguate & Prune pages:

- Prune by commonness $\leq \tau$

  (τ tradeoff: speed vs recall)

- Vote every candidate page, for both anchors

$$Vote_b(p_a) = \frac{\sum_{p_b \in Pg(b)} sim(p_a, p_b) \cdot P(p_b|b)}{|Pg(b)|}$$

a, b = anchors

$Pg(b)$ = all candidate pages of b

p_a, p_b = a candidate page of a, b

$sim(p_a, p_b)$ = relatedness between p_a e p_b

$P(p_b, b)$ = commonness of b in p_b

- Select top-E pages based on the vote value, for both anchors
- Select the best in commonness, for both anchors

Result....