Named Entity Recognition and Linking in Tweets based on Linguistic Similarity

Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone
(arianna.pipitone, giuseppe.tirone, roberto.pirrone)@unipa.it

Named Entity Recognition and Linking (NEEL) is a sub-task of information extraction that aims at locating and classifying each named entity mention in a tweet into the classes of a knowledge base, such as DBPedia.

According to [1], NEEL consists of:
- **mention detection** related to the identification of the entity mention in a tweet, and
- **candidate selection** related to the identification of the link in DBPedia that defines such an entity.

The figure shows the proposed architecture.

**Mention Detection**
The Chunking Activity module outputs all the possible words in the tweet to be analyzed for entity mention detection. The basic considerations are:

- Given a tweet $t$, its components can be classified in two categories based on the linguistic properties of the inherent chunks, that are:

  1. $M = \{m_i | m_i$ is a micropost of $t_i\}$, which contains both the main post that generates a discussion, and all the posts in the thread; chunks can be identified by blank spaces between words;
  2. $H = \{h_i | h_i$ is a hashtag or a tag of $t_i\}$, which contains the hashtags and the tags in $t$; chunking is not trivial, because no typical separation characters are used.

- **Informal language can influence linking:** the chunks devised so far must be rewritten using words already owned by the system. For this purpose, automatic correction [1] based on the WordNet source is applied to the identified chunks.

Formally, let be:

- $tok(s)$ the function that returns the list of tokens $L_s$ split using blank spaces for the string $s$;
- $a\_star(s)$ the function that returns the list of chunks $L_c$ for the string $s$ based on the $A^*$ strategy reported in [2];
- $ichl(k)$ the function that returns the list $L_{c_w}$ of the words that are syntactically similar ($\approx$) to the token $k$, using the automatic correction $^1$.

The Chunking Activity module implements the functions:

$$c_m : H \cup M \rightarrow L_c$$

$$c_m(s) = \begin{cases} a\_star(s) & s \in H \\ a\_star(tok(s)) & s \in M \end{cases}$$

$ichl : L_c \rightarrow L_w, ichl(k) = \{w_i | w_i \approx k\}$

whose output is the set $C = H \cup M \cup L_{c_w}$.

It is well acknowledged [1] that an entity in a tweet can be only a proper noun (NP or NPS), and a POS tagger $^2$ is applied to the words in $C$ for identifying the possible candidates to be a mention. The process ends with the definition of the set $MD$ that will contain all candidate mentions:

$$MD = \{m_i | m_i = \{c_i, c_{i+1}, \ldots, c_{i+n}\} \subset C, pos(m_i) \in \{NP, NPS\}\}$$

being $n_i$, value the extent of the $i$-th mention.

**Candidate Selection**
The Mapping to Meanges module from QuASIT [3] is adapted for candidate selection; the $a\_cneel$ function returns, for each mention in $MD$, the best matching entities in DBPedia:

$$a\_cneel(m_i) = \{ c_k | stem(m_i) = stem(i) \land sim(concat(m_i, i_j) > \tau, i_j = map(c_k), c_k \in C \}$$

where:

- $stem(w)$ returns the stem of the word $w$;
- $sim(w, w_j)$ returns the distance between two words by combining their Jaro-Winkler $^3$ and Levenshtein $^4$ distances:

$$sim(w, w_j) = 0.5*\text{jarowinkler}(w, w_j) + 0.5*\text{levenshtein}(w, w_j)$$

The $\tau$ value was experimentally fixed to 0.7 as better threshold for $sim(\cdot, \cdot)$.

- $concat(m)$ returns the chunks concatenation in a mention $m$;
- $map(c)$ returns set $\{i_j\}$ containing the instances in DBPedia whose stem of their class label is similar or equal to a mention stem in $MD$.

$C$ is the set of class names in DBPedia.

The set $\cup_{MD} a\_cneel(m_i) \cup I$ is the assertion graph of the tweet in DBPedia that realizes our NEEL task.

References

[5] https://courses.washington.edu/user/v02/permtable/