Entity Linking for Queries

zhhan@student.ethz.ch

Thursday 26th May, 2016

Method: Learning correct annotations

One problem of Baseline is it only uses *commonness* to disambiguate entities. A more reasonable way is adding contextual information (see example).Here, we consider disambiglation as a binary classification problem. And we can construct features that reflects contextual information using Word2Vec method. In this way, we can efficiently distinguish true and false annotations. Details are explained in next section.

Example output of Baseline: kubota bx 2200 tractor

Ground truth:	*System output:*
`kubota (0, 6)` $\rightarrow$ `kubota (1773523)`	`kubota (0, 6)` $\rightarrow$ `kubota (1773523)`
`tractor (15, 22)` $\rightarrow$ `Tractor (152692)`	`tractor (15, 22)` $\rightarrow$ `Tractor (152692)`
	~~`bx (7, 9)` $\rightarrow$ `BX (sternwheeler) (10864861)`~~

Method: learning with contextual features

- **Training data generation:** In the training data, we can get the queries $Q = \{q_1, \dots, q_n\}$, and true annotation set $\{a_{i1}, \dots, a_{ij}\}$ for q_i. Each annotation a_{ij} is a *mention-entity* pair, that is $a_{ij} = (m_{ij}, e_{ij})$. For every mention m_{ij} in ground truth, we use the first three entities returned by *WikiSense*[1] as its candidate entity set $\mathcal{E}^c$. Table 1 shows that only using the first three or five entities returned by *WikiSense* would be enough. Then we can lable each *mention-entity* pair (m, e) in $\mathcal{E}^c$ as positive class (+1, correct annotation) or negative class (0, false annotation).

- **Feature generation:** For each *mention-entity* pair (m, e) in training set, we construct its three features: (1) the *commonness* of (m, e); (2) contextual similarity of wikipedia title and remaining part of query except for this mention; (3) the contextual similarity of the first paragraph of wikipedia and remaining part of query except for this mention. The contextual similarity is computed using word embedding[2].

[1] We discard this mention if it is not in *WikiSense*

[2] the pre-trained embedding file could be downloaded from https://github.com/3Top/word2vec-api, https://github.com/stanfordnlp/GloVe.

Table 1: The rank distribution of true entity. We can find that most true entities in the candidate list returned by *WikiSense* is in top 3, e.g. $92\%, 94\%, 91\%$ in A, B and devel.

	1	2	3	4	5	others	totally found	no found
A	256	28	7	6	4	17	318	123
B	279	18	8	1	1	15	323	110
devel	255	28	9	5	3	20	320	99

Table 2: The result on GerdaqDevel

		mac			mic			std		
		p	r	F	p	r	F	p	r	F
Baseline	C2W	0.497	0.571	0.465	0.437	0.527	0.495	0.399	0.419	0.391
	Sa2W	0.475	0.544	0.441	0.412	0.540	0.467	0.400	0.422	0.390
Learning to Link	C2W	0.659	0.538	0.497	0.574	0.533	0.553	0.410	0.414	0.405
	Sa2W	0.634	0.510	0.472	0.538	0.501	0.519	0.418	0.415	0.405

- **Training classifier:** We use a simple binary classifier to learn the training data. Here, quadratic svm or random forest are used.
- **Entity linking in new data set:** For each query in the test set, we can use *WikiSense* to spot all linkable mentions. Then we construct features of each linkable mention with its 3 top entities returned by *WikiSense*. And we can find the correct entity for this mention using the trained classifier.