(TAGME/WAT) SMAPH-S with Entity-Merge and Stochastic Gradient Boosting

Simon Kassing, Ingo Schilken, Nikolas Kolitsas, Nily Ferrer

May 2016

1 Introduction

For the entity linking task, we decided to perform our own variants of SMAPH-S system [1] [2]. Based on the existing literature, a new implementation was written without going into the open source code available online. Our system provides three key contributions: (1) a novel approach to the ambiguity in the paper regarding candidate binding generation and features, (2) a re-interpretation and improvement of the concisely described machine learning approach, and (3) a novel entity merge phase (see Section 2.3).
Two pipelines are considered, one with TAGME [3] as candidate-entity generator; the other with WAT [4] via a public endpoint. The complete pipeline is displayed below in Figure 1.

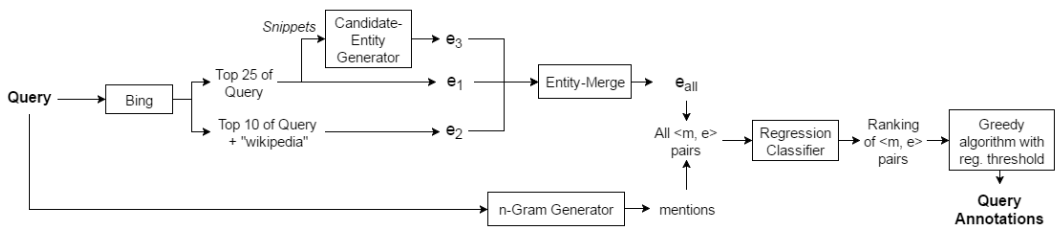

Figure 1: TAG+SM/WAT+SM Pipeline

2 Candidate-Entity Generation

2.1 Generators

Two generators were created for the generation of the candidate-entity set: (a) TAGME [3], re-implemented by us, specifically for the disambiguation phase we used the proposed voting scheme with disambiguation by threshold (DT) ($\epsilon = 30\%$ and $\tau = 2\%$) and for the anchor pruning phase we used the average method (AVG) between the link probability and the coherence ($\rho_{na} = 0.00$), and (b) WAT [4], called using the online WikiSense API exposed by the BAT framework. Both use the snippets of the Bing results to provide context for the query, and produce an expanded initial (semi-pruned) candidate-entity set.

2.2 Feature Implementation

The features described by the SMAPH-S system were generated, and adapted for both pipelines. Some features were impossible to implement exactly as described in the paper, or were infeasible, namely the following:

- **isNE**: the dataset that qualifies whether a Wikipedia entity is a named entity is no longer available;
- **avgRank**: in the case that there were less than 25 results, division is done by result size;
- **pageRank**: the Wikipedia information from the spotter regarding the entities only produces zeros (probably for legacy reasons) for fields such as PageRank, inDegree and outDegree.

2.3 Entity-Merge Phase

To add additional significance to entities that occur both in the $\varepsilon_{1,2}$ and ε_3, and to prevent a loss of information, we introduce a merge beyond the regular set-union operation to form ε_{all}. Namely, those overlapping entities' features are merged into one, preserving valuable classification information.

Annotator	C2W-$F1_{avg}$	C2W-$F1_{mi}$	A2W-$F1_{avg}$	A2W-$F1_{mi}$
SMAPH-1	52.1	55.9	/	/
SMAPH-S	51.4	55.8	46.3	50.2
SMAPH-2	54.4	57.0	51.4	53.2
SVR TAG+SM	52.5	55.7	49.0	50.7
SVR WAT+SM	54.2	58.0	50.3	53.2
Forest WAT+SM	55.1	59.1	52.1	55.5
Boost WAT+SM	55.1	59.8	52.6	56.7

Table 1: C2W/A2W results for all machine learning approaches on GERDAQ test (white are of [1], light-gray are our annotators).

3 Machine Learning

Optimization of the parameters and approaches is done, as in the original paper [1], towards the $F1_{avg}$ and $F1_{mi}$. For both pipelines, the parameters were kept the same as there was no significant preference in terms of resulting performance between the two candidate-entity generators. All parameter optimization was done on the GERDAQ development set. The greedy threshold is set respective to the regression value range each machine learning approach. Taking into account the class imbalance, the positive samples get a per-sample weight of 6 and the negative samples get a per-sample weight of 3. Missing feature values are replaced by the mean feature values, and the data is standardized (using the variance and mean from the training set).

3.1 ε-SVR

First, the entity pruning was done with the help of ε-support vector regression (ε-SVR). The ε-SVR model is trained on the GERDAQ training sets, using the features as described in Section 2.2. For the ε-SVR model, an RBF kernel is used (penalty parameter C=1, ε-tube=0.1). The regression threshold for classification is set to 0.14.

3.2 Random Forest

Second, a random forest regressor was used. Compared to the ε-SVR, this led to better results. The number of trees in the forest is set to 50 and the mean squared error is used to measure the quality of a split. The regression threshold for classification is set to 0.20.

3.3 Gradient Boosting

Third, stochastic gradient boosting was used with decision trees. This model outperformed the previous models, which is shown in Table 1 and Section 5.3 and 5.4 in the appendix. The number of trees in the forest is set to 100 and a least squares loss function is used. The regression threshold for classification is set to 0.20.

4 Experimental Results

Following the reasoning of [1], a focus is put upon the F1 scores. The complete C2W results are shown in Table 2, the complete A2W results in Table 3. Highlighted in light gray are our annotators.

Our annotators were trained to optimize the F1 scores, as it resembles a functional balance between the precision and recall. The TAG+SM pipeline performed worse than the WAT+SM pipeline, as it produced both less entities and with less accuracy. Therefor, further exploration into other machine learning approaches was only done for the WAT+SM pipeline.

With the machine learning improvements (Section 3), we were able to incrementally improve our results (see Table 1). Secondly, a contributing factor is the entity-merge phase (Section 2.3). Thirdly, another explanation could be a different interpretation of features (Section 2.2), which implicitly causes some potentially beneficial feature selection. In the end, we were able to create an annotator (*Boost WAT+SM*) to match (and even slightly surpass) the best results presented in the original paper (*SMAPH-2*).

5 Appendix

5.1 GERDAQ Critique

The GERDAQ dataset was annotated via CrowdFlower workers, of which the process was split into the recall-oriented and precision-oriented phases. A few problems arose from inspection: (a) annotations by workers are made on character-level instead of the tokenization level that is used by the SMAPH systems, (b) special characters, including quotes, are removed at the start and end of queries, but not in between, leading to strong query formulations such as transformation of [*"barack obama" and "michelle"*] to [*barack obama" and "michelle*], and (c) the definition of what is an entity varies wildly, leading to sporadically linking of mundane words such as *work* to the Wikipedia article of working.

5.2 Implementation and Execution

The pipeline implementation in Java follows the name-giving in Figure 1. The Bing queries have been cached, as there is a hard limit imposed by Microsoft and to improve speed. The main bottleneck lies for the TAG+SM pipeline in the voting for anchors in snippets in the generator (requiring many network requests). The WAT+SM pipeline is significantly faster, requiring only a single request. When the Python scripts converge is very dependent on given parameters. Final annotation and score calculation is quite quick using the Greedy approach.

5.3 Plain Results Output

5.3.1 SVR TAG+SM

```
C2W mac-P/R/F1:  0.622/0.577/0.525 mic-P/R/F1:  0.552/0.562/0.557
TP/FP/FN: 230/187/179 std-P/R/F1:  0.420/0.419/0.414
A2W-SAM mac-P/R/F1:  0.585/0.541/0.490 mic-P/R/F1:  0.496/0.518/0.507
TP/FP/FN: 212/215/197 std-P/R/F1:  0.422/0.416/0.409
```

5.3.2 SVR WAT+SM

```
C2W mac-P/R/F1:  0.667/0.580/0.542 mic-P/R/F1:  0.595/0.565/0.580
TP/FP/FN: 231/157/178 std-P/R/F1:  0.417/0.422/0.416
A2W-SAM mac-P/R/F1:  0.625/0.541/0.503 mic-P/R/F1:  0.540/0.523/0.532
TP/FP/FN: 214/182/195 std-P/R/F1:  0.426/0.421/0.415
```

5.3.3 Forest WAT+SM

```
C2W mac-P/R/F1:  0.674/0.595/0.551 mic-P/R/F1:  0.609/0.575/0.591
TP/FP/FN: 235/151/174 std-P/R/F1:  0.402/0.423/0.410
A2W-SAM mac-P/R/F1:  0.643/0.564/0.521 mic-P/R/F1:  0.570/0.540/0.555
TP/FP/FN: 221/167/188 std-P/R/F1:  0.410/0.423/0.410
```

5.3.4 Boost WAT+SM

```
C2W mac-P/R/F1:  0.662/0.611/0.551 mic-P/R/F1:  0.596/0.599/0.598
TP/FP/FN: 245/166/164 std-P/R/F1:  0.402/0.415/0.409
A2W-SAM mac-P/R/F1:  0.635/0.585/0.526 mic-P/R/F1:  0.564/0.570/0.567
TP/FP/FN: 233/180/176 std-P/R/F1:  0.409/0.419/0.410
```

5.4 Full Performance Tables

5.4.1 C2W: Strong Tag Match

Annotator	P_{avg}	R_{avg}	$F1_{avg}$	P_{mi}	R_{mi}	$F1_{mi}$
SMAPH-1	77.4	54.3	52.1	58.5	54.0	55.9
SMAPH-S	64.8	56.2	51.4	57.0	54.7	55.8
SMAPH-2	72.1	55.3	54.4	64.1	51.3	57.0
SVR TAG+SM	62.2	57.7	52.5	55.2	56.2	55.7
SVR WAT+SM	66.7	58.0	54.2	59.5	56.5	58.0
Forest WAT+SM	67.4	59.5	55.1	60.9	57.5	59.1
Boost WAT+SM	66.2	61.1	55.1	59.6	59.9	59.8

Table 2: C2W results on GERDAQ test, all entities.

5.4.2 A2W: Strong Annotation Match

Annotator	P_{avg}	R_{avg}	$F1_{avg}$	P_{mi}	R_{mi}	$F1_{mi}$
SMAPH-S	59.5	50.6	46.3	51.0	49.4	50.2
SMAPH-2	68.4	52.3	51.4	59.9	47.9	53.2
SVR TAG+SM	58.5	54.1	49.0	49.6	51.8	50.7
SVR WAT+SM	62.5	54.1	50.3	54.0	52.3	53.2
Forest WAT+SM	64.3	56.4	52.1	57.0	54.0	55.5
Boost WAT+SM	63.5	58.5	52.6	56.4	57.0	56.7

Table 3: A2W results on GERDAQ test, all entities.

References

[1] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. A piggyback system for joint entity mention detection and linking in web queries. In *Proceedings of the 25th International Conference on World Wide Web*, pages 567–578. International World Wide Web Conferences Steering Committee, 2016.

[2] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Hinrich Schütze, and Stefan Rüd. The smaph system for query entity recognition and disambiguation. In *Proceedings of the first international workshop on Entity recognition & disambiguation*, pages 25–30. ACM, 2014.

[3] Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1625–1628. ACM, 2010.

[4] Francesco Piccinno and Paolo Ferragina. From tagme to wat: a new entity annotator. In *Proceedings of the first international workshop on Entity recognition & disambiguation*, pages 55–62. ACM, 2014.