Best Matcher Explanation

Zhiwei Fan zfan29@wisc.edu
Lingfeng Huang lhuang58@wisc.edu
Fang Wang fwang64@wisc.edu

December 1, 2016

Description of Data Generation and Transformation

During the project stage 2, we have got around 70000 tuple pairs and observed that it was nearly impossible to get at least 50 positive sample points (tuple pairs that match) from the 400 sampled tuple pairs during the labeling stage, thus we have performed *reblocking* adding additional blockers and blocking rules.

Blockers Used in Re-blocking

Numeric Value Blocker:
Calculating the *absolute value* of *difference* between publishedYear, publishedMonth, publishedDay, pages as *yearDiff, monthDiff, dayDiff, pageDiff*. Any tuple pair that is found to satisfy ***any*** rules defined in {yearDiff≤2, monthDiff≤2, dayDiff≤2, pageDiff≤10} will be filtered out.

Isbn Partial Blocker:
Filtering out those tuple pairs got from the *numeric_value_blocker* that *do NOT* agree on the *first four digits of isbn*

Title Rule-based Blocker:
Filtering out those tuple pairs got from the previous two blockers described the jaccard similarity score of titles *smaller* than 0.5.

After conducting *reblocking*, we have finally got 5369 tuple pairs in our *new tableC.csv*

Labeling

After labeling and *throw away* those tuple pairs that we are not sure whether they match, we have finally got *350* tuple pairs with ***130 positive*** examples and ***220 negative*** examples in our *golden data* (tableH.csv).

Feature Construction

We have taken advantage of different *similarity measurement* functions in *py_entitymatching* (e.g., py_entitymatching.affine(s1,s2)) to construct both *sequence-based* (e.g., py_entitymatching.lev_dist(s1, s2),) and *set-based* (e.g., py_entitymatching.jaccard(arr1, arr2)) features on the original *string* attributes (title, author, publisher). We have also constructed features on original *numeric* features (publishedYear, publishedMonth) using py_entitymatching.rel_diff(d1, d2) and py_entitymatching.abs_norm(d2). *publishedDay* is not used due to its *high variance* even in *matching tuple pairs*. In total, we have constructed 42 features initially before performing any feature selection.

Development and Evaluation Sets

After Randomly split the set H (gold data) into a development set I of 250 examples and an evaluation set J of 100 examples, we got the I and J with the following statistical data:

Sets	Number of Positive Examples	Number of Negative Examples
Development Set I	97	153
Evaluation Set J	33	67

Initial Cross Validation Measurement

We have chosen several learning models and performed 10-fold cross validation on these models using the development dataset I. The corresponding ***initial*** (the first time) precision, recall and F-1 are shown as the following:

Learning Model	Precision	Recall	F1
Decision Tree	0.819	0.803	0.810
Random Forest	0.837	0.867	0.852
Linear SVM	0.742	0.821	0.779
Gaussian Naive Bayes	0.702	0.837	0.763
Bernoulli Naive Bayes	0.686	0.856	0.762
L1 Logistic Regression	0.729	0.753	0.741
L2 Logistic Regression	0.726	0.785	0.754

We have noticed that ***random forest*** gave the best performance considering *precision, recall* and *F1* together. However, the fact that F-1 score of *random forest* is only 0.852 is not very promising. Thus, we continued to *tune* the models and performed *feature selection* with the hope to improve F-1 score (as well as precision and recall) of *random forest*.

Debugging Iterations

We have first tuned *random forest* itself. The parameters we have tuned include *the number of trees, the maximal depth, the minimal sample split, the minimal samples leaf* and *the maximal features to consider*. Later on, we have performed *feature selection* with *backward selection* (each time reduce one feature and keeps the *subset* of features that the maximal *F-1 score* was observed). Two features *author_affine_score* and *publisher_smith_waterman_score* and have achieved 0.923 *precision*, 0.913 *recall* and 0.918 *F-1 score* in the *10-fold cross validation* process on *development set I*. Amazingly, on *evaluation set J*, we have seen 0.941 *precision*, 0.967 *recall* and 0.954 *F-1 score*. In order to get *stable* result during the *feature selection*, *10 iterations* of experiments have been conducted, hundreds of experimental data has been used. For space saving reason, we omit those tedious data report and do not think these *intermediate data* is necessary to be displayed.

Final Cross Validation Measurement and Measurement on Evaluation Set

Final Cross Validation Measurement

After tuning *random forest* and performing *feature selection*, we have re-run *cross-validation* on development set I on all the models and the results are shown as the following:

Learning Model	Precision	Recall	F1
Decision Tree	0.821	0.790	0.805
Random Forest	0.923	0.913	0.918
Linear SVM	0.736	0.832	0.762
Gaussian Naive Bayes	0.702	0.837	0.763
Bernoulli Naive Bayes	0.683	0.856	0.759
L1 Logistic Regression	0.731	0.767	0.748
L2 Logistic Regression	0.740	0.783	0.761

Measurement on Evaluation Set

Of course, we have *evaluated* our *final selected models random forest* along with *other models* on the *evaluation set* J and the corresponding results are shown as the following:

Learning Model	Precision	Recall	F1
Decision Tree	0.806	0.879	0.841
Random Forest	0.941	0.970	0.955
Linear SVM	0.769	0.909	0.833
Gaussian Naive Bayes	0.714	0.909	0.800
Bernoulli Naive Bayes	0.738	0.939	0.826
L1 Logistic Regression	0.634	0.788	0.703
L2 Logistic Regression	0.684	0.788	0.732

List of final features in final feature set

{_id, ltable_id, rtable_id, title_affine_score, title_lev_dist_score, title_lev_sim_score, title_jaro_score, title_jaro_winkler_score, title_needleman_wunsch_score, title_smith_waterman_score, title_jaccard_score, title_cosine_score, title_overlap_coeff_score, title_dice_score, title_monge_elkan_score,
author_lev_dist_score, author_lev_sim_score, author_jaro_score, author_jaro_winkler_score, author_needleman_wunsch_score, author_smith_waterman_score, author_jaccard_score, author_cosine_score, author_overlap_coeff_score, author_dice_score, author_monge_elkan_score,
publisher_affine_score, publisher_lev_dist_score, publisher_lev_sim_score, publisher_jaro_score, publisher_jaro_winkler_score, publisher_needleman_wunsch_score, publisher_jaccard_score, publisher_cosine_score, publisher_overlap_coeff_score, publisher_dice_score, publisher_monge_elkan_score,
year_rel_diff, year_abs_norm, month_rel_diff, month_abs_norm, pages_rel_diff, pages_abs_norm}

Approximate Time Estimation

It took us approximately 6 hours to finish *reblocking* and *labeling*. We have spent most time *tuning* models and *perfoming feature selection*. A big amount of experiments have conducted for us to detect the problems and give ideas of model tuning. The experimental stage took us approximately 10 hours. In total, approximately 20 hours have been spent on the whole process.

Discussion

Since we have chosen the domain *book*, it is very hard to come to an agreement on *the standard of matching*. Books of different versions should be regarded as *non-match*. But during the *labeling* stage, we have observed other factors that could significantly influence the *matcher learning* and essentially *matching* in a *negative* way. For example, the same book could be published by different publishers during totally different time; the book that has multiple authors could have *different author information* on different source websites: one website lists partial of the authors of the book while the other website lists the rest. These *inconsistency* has greatly impacted the *learning* process and prevent us from improving the quality of matcher to higher standard (with higher precision, recall and F-1).