An Ensemble Approach to Financial Entity Matching for the FEIII 2016 Challenge

Enrico Palumbo
ISMB, Italy
palumbo@ismb.it

Giuseppe Rizzo
ISMB, Italy
giuseppe.rizzo@ismb.it

Raphaël Troncy
EURECOM, France
raphael.troncy@eurecom.fr

ABSTRACT

Financial entities are often referred to with ambiguous descriptions and identifiers. To tackle this issue, the Financial Entity Identification and Information Integration[1] (FEIII) Challenge requires participants to automatically reconcile financial entities among three datasets: the Federal Financial Institution Examination Council[2] (FFIEC), the Legal Entity Identifiers (LEI) and the Security and Exchange Commission[3] (SEC). Our approach is based on the combination of different Naive Bayes classifiers through an ensemble approach. The evaluation on the Gold Standard developed by the challenge organizers shows F1-scores that are above the average of the other participants for the two proposed tasks.

Keywords

FEIII, financial entity matching, Naive Bayes, ensemble, union voting, majority voting

1. APPROACH

The first step in our approach is to select a set of properties $i = 1..K$, such as name, address or zip code, on which the comparison between two entities has to be performed. Then, in order to perform matching only with relevant candidates and avoid the naive $O(n^2)$ implementation, we reduce the search space using an inverted index, in which property values are the indexes and the tuples are the documents referred by the indexes. Each entity e_1 is compared with a candidate matching entity e_2, computing a set of similarity scores s_i for each property. These values are combined to come to a final decision through a Naive Bayes classifier, using the rule popularized by Graham's spam filter[4] (details of the derivation can be found in [1]):

$$Match \iff \frac{p_1 p_2..p_k}{p_1 p_2..p_k + (1-p_1)(1-p_2)..(1-p_k)} > t \quad (1)$$

where $p_i = P(Match|s_i)$. The derivation of the formula is based on two key assumptions. First, we assume that, *a priori*, $P(Match) = P(NoMatch)$. Second, we assume that features s_i are independent. The decision rule has then a very intuitive interpretation. A pair of records is considered to be a match if the probability that it is a match given the set of observed similarity scores is above a certain threshold. The threshold t has to be defined experimentally and governs the trade-off between precision and recall of the algorithm. If $t = 1$ the Eq. 1 can never be satisfied, then $p = 1$ and $r \to 0$. On the contrary, if $t = 0$, $p \to 0$ and $r = 1$. In the absence of a training algorithm, the user has to manually tune the threshold t, favoring precision over recall or vice versa, or finding a good balance between the two, depending on one's needs. Following the empirical evidence that ensemble learning can improve entity matching systems [2], we create an ensemble of Naive Bayes classifiers and then combine them:

1. Start from a Naive Bayes classifier $f(e_1, e_2; t)$
2. Generate an ensemble of classifiers $f_i(e_1, e_2; t_i)$ with $t_i = t + aU$ and where U is a real random number uniformly distributed in the interval $[-0.5, 0.5]$ and a is the total maximum amplitude of perturbation
3. Combine the decisions into a final decision $F(f_i(e_1, e_2; t_i))$

The intuition behind this approach is that combining the decisions of an ensemble of classifiers enables to overcome the trade-off between precision and recall that is introduced by Eq. 1.

2. CONFIGURATION AND RESULTS

In this section, we describe the configuration and the results that we have obtained for the Task 1 (FFIEC → LEI) and Task 2 (FFIEC → SEC) proposed by the challenge. Our implementation is based on Duke[5] and on a collection of Python scripts that combines Duke's outputs to perform ensemble classifications. In order to tune the algorithm, we have built a Gold Standard through a semi-automatic procedure, in which we have kept a low threshold t to maximize the recall and then manually checked the matching entities. Duke is built by default on top of a Lucene Database[6] which indexes the records through an inverted index and can be

[1] https://ir.nist.gov/dsfin/about.html
[2] https://www.ffiec.gov/
[3] http://www.sec.gov/
[4] http://www.paulgraham.com/spam.html
[5] https://github.com/larsga/Duke
[6] https://lucene.apache.org

DSMM'16, June 26-July 01 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2951894.2951906

FFIEC	LEI	SEC	Property	Cleaner	Comparator
Financial Institution Name Cleaned	Legal Name Cleaned	CONFORMED-NAME	Name	LowerCaseNormalize + FinancialInstitutionName	Semantic Financial Institution
Financial Institution Address	Legal Address Line Cleaned	B-STREET	Address	LowerCaseNormalize	JaroWinkler
Financial Institution City	Legal Address City	B-CITY	City	LowerCaseNormalize	JaroWinkler
Financial Institution Zip Code	Legal Address Postal Code	B-POSTAL	State	LowerCaseNormalize	Exact
Financial Institution State	Legal Address Region 2	B-STPR	Zipcode	DigitsOnly	Exact

Table 1: Configuration of properties, cleaners and comparators for each dataset

configured by setting a number of parameters such as the **min-relevance**, namely a threshold for Lucene's relevance ranking under which candidates are not considered. We have empirically set **min-relevance** = 0.7 for Task 1 and **min-relevance** = 0.4 for Task 2. The choice of properties, of cleaners and comparators is reported in Tab. 1. In addition to Duke's cleaners and comparators, we have defined our own Duke's cleaner[7], which allows to expand common financial abbreviations such as "fsb" → "federal savings bank", "inc" → "incorporated", "co" → "company", "corp" → "corporation" and the like. Moreover, we have defined a specific comparator[8] that aims to improve the precision by allowing a match only if certain keywords appear in both entity's names. For instance, we have observed that a common false positive was considering the same legal entity for the holding of a bank and the bank itself, such as "Isabella bank" and "Isabella bank corp". Thus, the `SemanticFinancialInstitutionComparator` allows a match only if keywords such as "corporation" are present in both names.

Method	t	a	N	p	r	F1
Duke	0.890			95.45	80.44	**87.31**
Duke	0.895			96.29	78.43	86.44
Ensemble Majority	0.830	0.02	10	96.46	77.02	85.65
Ensemble Union	0.830	0.02	10	96.32	79.23	86.95
avg				80.49	86.90	79.78

Table 2: Results in % for Task 1 (FFIEC-LEI)

In Tab. 2, we report the results for Task 1 and in Tab. 3, those of Task 2, for different configurations of the threshold t and of the combination rule F (the number N of classifiers and the amplitude a of perturbation have been kept constant). More specifically, "Majority" refers to majority voting, where a pair of entities is considered to be a match if more than a half of the configurations considers it as a match, whereas "Union" refers to union voting, where a pair of entities is considered to be a match if at least one of the configurations considers it as a match.

3. ERROR ANALYSIS

We observe that the system works well for both tasks, yielding F1 scores for all configurations above the average in comparison to the other submissions. We also obtain better results for the Task 1 than for Task 2. The algorithm appears to be unbalanced, though, favoring precision over recall. This is due to the fact that we have tuned the algorithm on the Gold Standard that we had created and that was probably missing true positives, leading to a biased recall estimation. As the case of having at disposal a complete Gold Standard to train the algorithm is quite uncommon, a general advice is to try to keep into account this likely bias by favoring recall over precision in the tuning phase. The use of exact comparators on the "State" and on the "Zip code" property is motivated by the fact that, having these two fields in a well defined format, they should include less ambiguity and misspellings. In spite of this sensible observation, we have observed that this choice is too strict, as there are cases of matching entities with different zip code. This is the major cause of the low recall of our algorithm.

Method	t	a	N	p	r	F1
Duke	0.870			86.67	56.52	68.42
Duke	0.865			82.21	58.26	68.19
Ensemble Majority	0.865	0.01	10	86.18	56.96	68.59
Ensemble Union	0.865	0.01	10	84.81	58.26	**69.07**
avg				63.86	71.80	62.36

Table 3: Results in % for Task 2 (FFIEC-SEC)

The ensemble methodology is generally increasing the performance of the algorithm, being able to raise both precision and recall at the same time, in accordance with our intuition. In particular, the union voting appears to be the best performing approach, as it increases the recall through a decision rule less strict than the majority voting. Furthermore, in a later work, we have realized that the performance tends to increase as the number of configurations N and the amplitude of perturbations a raises. However, this also increases the computational time, as it requires to run Duke N times. In a later work, we have implemented stacked generalization on top of the Naive Bayes classifiers and we have applied less strict comparators, obtaining significantly better performance[9].

4. REFERENCES

[1] E. Croot. Bayesian spam filter. http://people.math.gatech.edu/~ecroot/bayesian_filtering.pdf - Last visited: 28 April 2016.

[2] H. Zhao and S. Ram. Entity identification for heterogeneous database integration, a multiple classifier system approach and empirical evaluation. *Information Systems*, 30(2):119–132, 2005.

[7] FinancialInstitutionNameCleaner.java

[8] SemanticFinancialInstitutionComparator.java

[9] https://github.com/enricopal/sfem