De-anonymization on Authors using Coauthor List

Fang Zhang
Computer Science, Tsinghua University
thuzhf@gmail.com

ABSTRACT

This paper describes how I conduct the de-anonymization on authors using their coauthor lists as features. The main model I use here is Multi Layer Perceptron model (MLP model). The authors and their coauthors I use to train and test are extracted from two conferences from the DBLP-Citation-network-V7[1] dataset. [2] I use 5-fold cross validation to test the dataset. And the average precision, recall, and F1 score I get on the test dataset are all over 98%.

1. INTRODUCTION

Assume that an author A has published several papers on both SIGKDD and ICML, there's chance that A's coauthors on SIGKDD are very similar with A's coauthors on ICML. If this is true, we can train a model based on this feature, and given any two authors' coauthors list, we can check whether they are the same author.

2. DATASET

I extract 3 pairs of conferences from the above dataset: SIGKDD-ICDM, SIGMOD-ICDE and NIPS-ICML. In every conference pair, I extract all authors who has published at least one paper on both conferences. And for every author, I extract his
her coauthors list and use this as features.

3. FEATURES

The features I use is every pair of authors' common coauthors list. To be specific, if author A and author B share a common coauthor C, C will be in their common coauthors list, and we can assign value 1 or somthing similar to this feature. And in my experiment, I firstly calculate the tf-idf value of every author's coauthors, for example, if author A has a coauthor B on SIGKDD, and A has published N_{AB} papers with B on SIGKDD, but B has published N_B papers with others on SIGKDD, and there're N_{ALL} authors on SIGKDD, then B's value V_B in A's coauthors list is calculated as follows:

$$V_B = N_A * log\frac{N_{ALL}+1}{N_B} \tag{1}$$

For two authors A in SIGKDD and B in ICDM, both A and B have a coauthors list in which every value is calculated as above. Then we can merge the two list into one list to form an input to our model, and value in each position of the merged list is calculated as follows: let A_i and B_i denote the corresponding value of the ith position in the two coauthors list, and C_i denotes the value in the final list of the same position, then we have:

$$C_i = \frac{2}{\frac{1}{A_i} + \frac{1}{B_i}} \tag{2}$$

That's to say C_i is the hamonic mean of A_i and B_i. We do this because we want that if one of A_i and B_i is too large compare to the other one, then C_i won't be too large.

4. CONSTRUCT POSITIVE AND NEGATIVE INSTANCES

For the same author in two conferences such as SIGKDD and ICDM, it is a positive instance. And in this experiment, I choose the same amount of negative instances as positive instances randomly. To be specific, for every author in SIGKDD, I choose another author which is not the same as this author in ICDM randomly, and use it as a negative instance.

5. MODEL

I use MLP model to train and test the dataset. My model's architecture is 5671x100x64x2, which indicates our input feature is 5671-dimensional, we have two hidden layers, the first hidden layer has 100 perceptrons, and the second hidden layer has 64 perceptrons. Our last layer is a logistic regression layer which has two outputs. The activation function in both hidden layers is sigmoid function. It is because there're 5671 different qualified authors in SIGKDD and ICDM that our input feature also has so many dimensions, and this will change if we choose other pair of conferences.

6. CROSS VALIDATION

I use 5-fold cross validation to train and test the dataset, and also to decrease overfitting.

7. TOOLS

I use TensorFlow [1] to construct the model and write code in Python.

8. RESULTS

I test the model on three pair of conferences: SIGKDD-ICDM, SIGMOD-ICDE, and NIPS-ICML. And I calculate

[1]Website: https://aminer.org/citation.

Table 1: Test Results

	SIGKDD-ICDM	SIGMOD-ICDE	NIPS-ICML
Precision	99.15%	99.69%	99.68%
Recall	98.06%	98.71%	98.82%
F1	98.60%	99.20%	99.25%

Table 2: Parameters

learning rate	0.001
num of iterations	1000
batch size	100
num of hidden layer 1 perceptrons	100
num of hidden layer 2 perceptrons	64

the precision, recall and F1 score on all these three pairs. The test results are in table 1. We can see that every score is over 98%, which suggests that our model has achieved quite good results.

9. DISCUSSION

9.1 Name Disambiguation

In this experiment, I simply treat two authors with the same name in two conferences as the same person. And this is obviously not always correct but I it's hard to come up with another solution for me now. And if we can solve this problem better, maybe we can get better results too.

9.2 Choosing parameters

When I firstly train the model, I set learning rate to 0.1 and I get much worse results, then I adjust the learning rate and other parameters on and on, finally I try following parameters in table 2 and get the above much better results. This kind of work is nonautomatic and tedious. And I think it is worthy for us figure out some kind of automatic parameters-adjusting methods in the future, which should be much better than our manual adjustment.

10. REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: Extraction and mining of academic social networks. In *KDD'08*, pages 990–998, 2008.