LARGE SCALE ENTITY RESOLUTION IN CO-AUTHORSHIP NETWORKS

Joel Castellon
LCA-3, LCA-4

Supervisors:
Lyudmila Yartserva
Brunella M. Spinelli

Professors:
Matthias Grossglauser
Patrick Thiran

Bachelor Semester Project

June 2015.

Abstract

State-of-the-art methods for entity resolution do not make a distinction between detection and resolution. Perhaps because they were designed as a general framework. Yet, a quick survey on the activity by the data management management team working on one of the largest bibliographic datasets (DBLP [13]) provides us with a different motivation. Despite having privileged resources, such as the paper corpus, meta-information, and technical competence for implementing the state-of-the-art, the common practice is far from being sophisticated. This is mainly because high complexity in the implementation and inefficiency in running time of state-of-the art approaches.

Nonetheless, the presence of synonyms and homonyms are possibly the two most important problems for data quality control DBLP team faces. In this project we address the homonymity problem only, this is because such problem is (in practice) easier understood in the language of graphs [11]. Therefore, we relax the way entity resolution is proposed for homonyms, this relaxation will be *detection*. We claim, by our results, and real practice of our study case, that detection by itself might be effective enough to solve the needs of many large datasets. In other words, in real practice, and for many types of networks, if one can detect homonyms then disambiguation can even be done by hand.

We start with an empirical study on the heuristic for homonym detection that the DBLP team uses day to day. We use it as an starting point to learn about the structural characteristics of homonym nodes. We study experimentally and analytically important aspects of the homonymity phenomenon as a process is the co-authorship network. We then extend DBLP's efforts for developing a fully generic and efficient approach, that is best suited for large networks.

1 Introduction

1.1 Homonyms and synonyms

Entity Resolution is the problem of disambiguating latent entities given observables in a dataset. In this context, disambiguation has two meanings. The first one is solving the synonyms problem, which consists of the occurrence of more than one instance of the same entity (author in our case) under different aliases. The second meaning, and the one that we address in depth in this particular project, is solving the homonyms problem. The homonyms problem arises when we have different entities under a single alias.

1.2 Motivation

This work has two main sources of interest. The first one is quality control, specially in datasets where homonymity and synonymity are quite common, and of enough relevance for information consumers. In this project we have worked specifically with the DBLP dataset [13] which is one of the largest bibliographic references for researchers in Computer Science and related fields. Entity resolution is also of high importance in other settings, in news articles or blogging networks for example. People write about trending topics and collisions of slightly variating titles can produce homonyms. Another case is that of online commerce websites that act as an intermediary for various retailers. Inventories can be very large, and products with the same purpose vary by very specific features even within a same brand. Collision can happen despite these products having a unique identifier or production number by the manufacturer, and this is because of human input error. A similar situation happens in DBLP, where unique identifiers are given to authors when they join the network. Nevertheless, one can never be sure of the uniqueness of the name under which every author publishes.

The second source of interest concerns information producers and analysts, the majority of which completely ignores in the curation steps as part of the analytics pipeline. The importance of the distortion is, of course, dataset dependant. Yet, for data extracted from network structure the bias can be significant.[9]

Outline We begin this report by explaining our methodology, in particular we state clearly which were the limitations on resources we had (Section 2). The goal on doing so, is to convey how simple is to implement our method. Its robustness to these limitations is validated by a list of actual twenty homonyms unknown to DBLP we have found. We explain how difficult is for DBLP to get even **one** true new homonym in Section 3. In Section 4 we make an empirical study of the DBLP method on canonical network topologies.

We have practical evidence that our method can detect homonyms much more frequently than DBLP's current expectations (Section 7). Improvements and extensions are possible, we make some suggestions in Section 8. Section 5 explains how we identified a good set of features. We even have provable guarantees for the clustering coefficient being a relevant feature. This also shows some caveats when handling dense homonym neighborhoods (Chinese names problem).

We want to point out that even if we use classifiers, this is not exactly

a classification problem. It could be more correctly defined as an *anomaly detection problem*.

Finally, the Discussion section (Section 8) points out the kinds of data sets where this approach is more promising. At the end of that section, we elaborate on qualitative characteristics of homonyms based on our experimental findings. This validates a-priori intuition on how homonyms appear and suggests some more.

2 Data acquisition and methodology

Our analysis evolved around a single data source: the co-authorship graph. Even if we had a lot other valuable sources of information at our disposal we did not use them at all. For example, there was the meta-data in the DBLP records: dates, affiliations, even references to homonyms when these had been disambiguated. Neither we looked at the papers' corpus. Our goal was to investigate how well we could do only based on network structural information. This clearly has its benefits in respect to the simplicity of implementation (Section 3). The parser we used was adapted from a Java parser the DBLP team had published around 2005, and that they call a *basic* parser.

We also had a limitation in computational resources. Using more powerful computers, would be an ideal situation when determining the optimal number of communities in the whole co-authorship graph. As it would be when generating a higher percentage of collisions in our simulations. Experiments were run in a personal computer, and it does not compare to the infrastructure available to professionals.

Our method is suitable for large scale data as it finds homonyms in any part of DBLP, at any state: we haven't constrained ourselves to conferences, or times slices to reduce complexity.

3 The landscape of the homonyms problem, and the DBLP approach

It is known that approaches that have tried to make inference based on records metadata had a really biased set of assumptions [7]. Michael Ley himself (the creator of DBLP) says in his paper [1] that DBLP has many ad-hoc solutions encrypted in its records. Some came about from shifts in storage technology, others as lack of information about certain populations (Asian names, for example). Beyond a basic set of common attributes shared

by most records, the structure these XML record is really non-homogeneous [1].

One can then wonder, if the DBLP team has privileged knowledge about this data-set's structure and even external resources of information (papers' corpus). Then why is that they use simple calculations in the co-authorship graph for homonym detection while neglecting all other available data.

3.1 The DBLP method.

The whole dataset has roughly 1.4 million authors, and the fraction of known homonyms (explicitly marked with numbers e.g. John Doe 0001, John Doe 0002) is *extremely* small. Contracting all homonym aliases we only obtain 1808 homonym examples out of 1.5 million authors. This is 0.12% of the total amount, even if we had 5% of the network as homonyms would make detection much easier. This is why it is challenging to train a good classifier. Now, in respect to DBLP's activities, the vast majority of homonym resolutions happen because of personal complaints, some of the authors affected send a notice. The best way to illustrate this is with a quote from [1]:

There is a daily job which looks for homonym persons with different ID numbers and a shared coauthor. Until now there were no homonyms close enough to share a coauthor. All alerts produced by our program were caused by input errors, we are still waiting for the first false hit... Again the fine tuning of this algorithm remains the challenge. Often the human user sees partial hits our algorithms are not able to locate.

The main DBLP algorithm is really simple. For every candidate homonym u take N_u (set of neighbors of u), these will be the witnesses for the identity of u. Now, if the induced subgraph on the neighborhood only $G_u = G[N_u]$ is connected we say that u is not an homonym, otherwise it is an homonym. We discuss this algorithm further in the following sections.

3.2 Classic entity resolution and the state of the art

In this section we'll discuss some of the reasons that make stat-of-the-art entity resolution impractical in DBLP, we'll comment based on our experience during this project. We know that it is in the best interest of the DBLP team to have very good automatic homonym detection. It is certainly not a matter of difficulty in the acquisition of external data. If they thought topic modelling would solve the homonyms problem, they could. They have probably one of the largest collections of papers in the world, they even provide such a service when asked. Then, why don't they try these methods

that have proven, in research, to be the most succesful [8]. We elaborate on an example typically shown on entity resolution papers.

The classic picture of entity resolution is as follows: there are two graphs involved, the observable graph and the knowledge graph that one adheres to the observable part in order to make inference. One can think of it, from example when disambiguating news articles, and the topics that involves these articles form the entity graph. While this should be feasible in DBLP is clearly not practical. If one thinks of the composed graph as a graphical model then the knowledge graph are the evidence variables, and the observed graph the ones for which to determine the state. Here, one does not have a topic linked to 200 articles as a knowledge reference. In a graphical model, this would imply the evidence variables to instantiate in the majority of inferences we to execute. It would be great to have evidence variables with dense edge connections to the majority of our observable graph, then yes, graphical models are ideal. However, efficiency in time is still the main obstacle for a periodic monitoring of an entire dataset. The difficulty for entity resolution of authors is not only time complexity, but in the formulation of the problem at such large scale. This is because, DBLP's co-authorship graph is like a knowledge graph itself. There is no trivial, nor automatic way way to know what is the real name of each author. The number of possibilities is becomes large, and there's not as much available information as for trending topics in a news articles network. While normalization, and similar procedures can be used to setup such knowledge graph when restricted to a subset of authors. The observed and knowledge graph are of similar size, and setting up the edges depends on delicate processing of these strings. Which is not the case for news articles, throwing edges to irrelevant topics for an article variable might not be of such negative impact. This is because, topics are normally clearly defined because everyone talks about them. Unfortunately, only few authors get such privilege.

With our approach, we succeed to reduce the number of candidate homonyms to the extent where it can be disambiguated manually. Once a homonym is detected, we look at papers' titles or conferences. This is normally enough to see there's more than one person in one such record (Section 5). Once detection is done, of course these sophisticated methods should do a much more efficient job than humans. We indeed appreciate the solutions that Getoor et Al. have provided to the entity resolution problem, but it is evident that in many real datasets it is not practical. As L. Getoor said herself in this conference [12], *there's not enough attention put to feature engineering* for graph mining. That is exactly what we were devoted to in this project, and we are very confident that our view and the classic one on

the entity resolution problem could work great in complement.

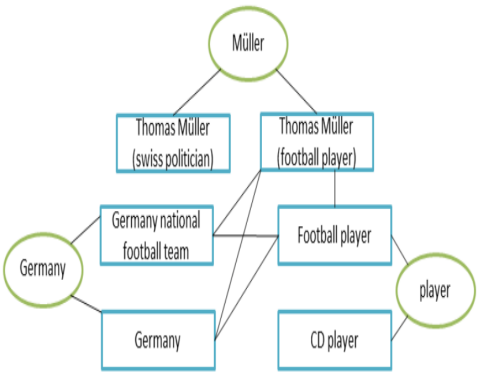


Figure 1: Example of a knowledge graph (green) together with the observable graph (blue) in classic entity resolution approaches.

4 Building intuition on the structural properties of homonyms

We started our analysis was with an empirical study on the properties of the DBLP method on graphs for which we could tune the parameters. A priori it is really difficult to make precise assumptions on how homonyms originate. So we make none, and the way we introduce homonyms in the networks we generate (and on DBLP for some experiments later on) is by colliding pairs of vertices chosen at random. One of the vertices takes the id of the paired one, and all former edges are aggregated in the resulting node. The size of the network in our simulations is really not important, as we study very local properties of nodes. Besides that, keeping the graph size small allows us to observe the effects of network contraction much more dramatically with respect to what we would have on real networks.

We studied three network models, $G(n, p)$, Preferential Attachment and Small World (Watts Strogatz). The last two being particularly insightful. We look at the evolution of the clustering coefficient and at precision and recall scores for the DBLP classifier. At every step of the simulation, the mentioned measures are taken for every vertex present in the graph.

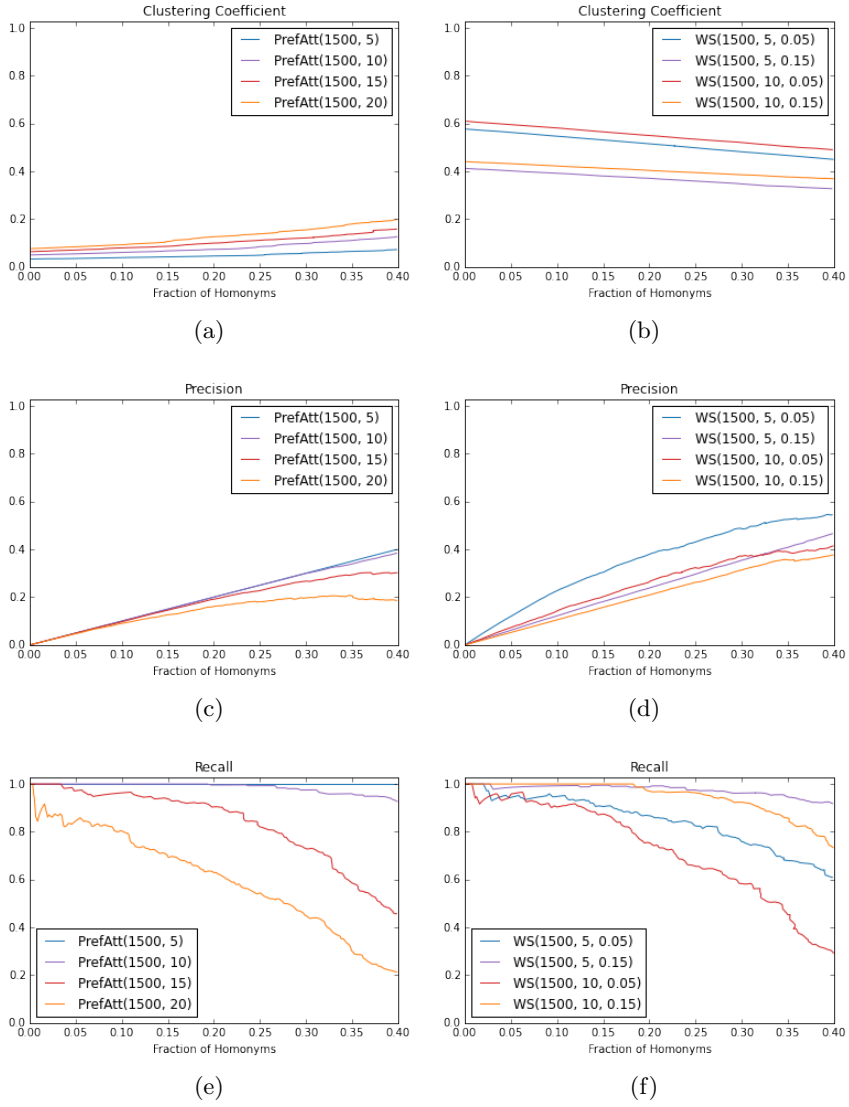


Figure 2: Left : Preferential Attachment(Nb. of nodes, expected degree). Right : Watts Strogatz(Nb. of nodes, rewiring probability).

4.1 Tackling the chinese names sub-problem

The chinese names subproblem is when homonym neighborhoods are quite dense. This is because it is not rare for an instance of this problem to have five to six names under a single record. This and the fact that chinese names are particularly keen to appear as homonyms (collaborators) produces dense homonym neighborhoods. Our results provided us the intuition to tackle the Chinese names problem, and understand why DBLP's method would overlook Chinese names homonyms. Our method is really successful at detecting Chinese problem instances, most of the examples we present in the Appendix are examples of this. First, we observe from Fig.:2(a) that the aggregate clustering coefficient increases monotonically, and it does so more rapidly for instances where there are higher degree nodes present. We can explain this as follows, imagine two high degree vertices a and b. The clustering within this model will always be low, as it is hard to induce so many edges in the neighborhood of a and b. However, when we collide vertices at random we are intersecting a and b's neighborhoods and making them smaller at the same time. The presence of these high degree vertices means more chances to close a and b's triangles. Say a has a high degree neighbor c , it probably closes many triangles for a and has lots of neighbors in common with b. When we collide c with a neighbor of b, now c closes many triangles for both.

The sudden drop in recall and precision is really simple, and is really the essence of the Chinese problem. The DBLP method detects homonyms when the egonet of a candidate breaks into components. As we saw with the clustering coefficient, the triangles in the neighborhood for in the presence of high degree nodes closes more quickly. Therefore, the DBLP classifier just overlooks true homonyms, whose egonet would break otherwise. An illustration for this situation, is that when prolific author with another one, and both working in different subjects really throws a lot of spurious edges in both authors neighborhoods.

4.2 The clustering coefficient as an important feature.

The feature that has interested us about W.S. model is high clustering coefficient (clcff), lower rewiring probability means higher clustering coefficient. As we see from Fig.2(b) the curve in this case is monotonic decreasing. The motivation for measuring the clustering coefficient came from an excellent study on the impact that similar perturbation strategies had on node-level measures [2]. This study suggested, that in the majority of models, the clustering coefficient is the most sensitive node-level measure for false aggregation. The avarage clustering coefficient in our samples from DBLP is 0.65 (Section 5).

The importance of the clcff as a feature is supported by the pronounced concavity in the precision curve Fig.2(d) for those instances that have initial high clustering. In fact, the unknown homonyms we present were obtained by a classifier were we tune the parameters to focus on precision. The clcff is, individually, the best feature we know about for homonym detection. Our method combines it with a few others but, in essence, this is the killer feature. The clcff is actually a soft version of the DBLP method. Namely, for a node u all of its co-authors have collaborated with each other, this means that $G[u]$ is a clique. Having a clique of co-authors is the higher confidence you can get from this structural feature. Then, we think of the clustering coefficient as measure of confidence from 0 to 1.

4.3 The relationship between the clustering coefficient and collisions.

The following lemma provides a theoretical reference on the suitability of the clcff to our purposes. The first three parts are very simple, yet concern the most common case and provide good measurements. This is because the gap between homonyms and non-homonyms with respect to the clcff gets bigger.

The fourth part is a bit more subtle. It provides a practical bound on the state of a node's neighborhood (see α in the lemma) for it to be detected as homonym. This threshold gives us an indication on how far we can push our detection method in the Chinese names problem. We use the contrapositive form of this statement. For our purposes $C_u \leq 0.6$ (see experiments), this means that if $\alpha_u > 0.6$ e.g. more than half the neighbors of u have false edges incident on them, we call suh neighbors spurious vertices. Then, the clustering coefficient can decrease only, so the gap gets smaller and we might not be able to differentiate non-homonym from homonyms.

The last part of the lemma can be a good indication on when to use different features. As a reference for what follows, homonyms have low clcff, while hon-homonyms have this measurement high (Section 5).

Lemma 1. *Let u_1 and u_2 be two colliding vertices and $N(u)$ denote the neighborhood of vertex u e.g. $N(u) = \{u : (u, v) \in E\}$. $(V, E) \to (V', E')$ the graphs before and after collision, C_u and C'_u respectively the clustering coefficient before and after collision for u. And, $N_e^u = |\{(x, y) \in E : x, y \in N(u)\}|$.*

- ***i)*** $|E'| \leq |E|$
- ***ii)*** $\forall x \notin N(u_1) \cup N(u_2) \cup \{u_1, u_2\}$ *we have* $C_x = C'_x$.
- ***iii)*** $x \in N(u_1) \setminus N(u_2)$ *then* $C_x \leq C'_x$.
- ***iv)*** *If* $x \in N(u_1) \cup N(u_2), C_x \leq C'_x$ *and* $N_x > 2$ *then* $\alpha_x \leq C_x$.
  where $s = |\{t \in V : t \in N(x) \cap N(u_1) \cap N(u_2)\}|$ *and* $\alpha_x = \frac{s}{N_x}$.

Proof.

- **i)** Let us call u the result of the collision of u_1 and u_2. Any edge $(u_1, x) \to (u, x)$, same for u_2.
  - (a) If $\exists x$ s.t. (u_1, x) or $(u_2, x) \in E \to (u, x) \in E'$.
  - (b) If $(u_1, u_2) \in E \to$ it contracts in E'.
  - (c) This is the only change in the edge set.
- **ii)** No node in $N(x)$ collides $\Rightarrow$ $|N(x)|$ stays constant.
  $\forall z, y \in N(x), (z, y) \in E' \iff (z, y) \in E$ because $z \neq u_1, u_2$ and $y \neq u_1, u_2$.
- **iii)** by *i)* $\forall y$ s.t. $(x, y) \in E$ and $(y, u_2) \in E \to (y, u) \in E'$.
- **iv)** Let $k = s + \mathbb{1}_{\{(C_1, C_2) \in E\}}$ be the number of edges between neighbors we loose upon collision.
  The clusering coefficients, before and after collision respectively:
  $C_x = \frac{2N_e^x}{N_x(N_x-1)}, C'_x = \frac{2(N_e^x-k)}{(N_x-1)(N_x-2)}$.
  We manipulate the inequalities in order to compare C_x and the number of spurious vertices in $N(x)$ (e.g. s).
  $C'_x \geq C_x \Rightarrow \frac{N_e^x-k}{N_x-2} \geq \frac{N_e^x}{N_x} \Rightarrow \frac{k}{2} \leq \frac{N_e^x}{N_x} \Rightarrow s \leq k \leq \frac{2N_e^x}{N_x}$
  $\Rightarrow \alpha_x N_x(N_x - 1) \leq \alpha_x N_x^2 \leq 2N_e^x \Rightarrow \alpha_x \frac{N_x(N_x-1)}{2N_e^x} \leq 1 \Rightarrow \alpha_x \leq C_x$

□

5 Speaking the language of graphs, feature engineering inspired by other methods.

In this section we describe the process of crafting relevant features for homonym detection. Some of these features were obtained by exhaustive examination of all measurements we could think of. While others are loosely inspired by more sophisticated methods that can be used in homonym detection, this is the case of how we used community detection by thinking about topic modelling. This is why we see feature engineering as a translation of powerful methods into more intuitive and easy to implement measurements e.g. the language of graphs.

The feature engineering approach we took was rather an intuitive one. This is actually a legitimate and widely studied technique. Specifically, it belongs to a set of problem solving techniques described as *transfer learning*. This area induces really deep research questions, that are at the core of some of the most ambitious projects in network analysis [6]. We keep it as a reference to understand why our approach scales well, the authors of [5] describe some strategies for extracting features that have inspired us. Among the mentioned applications of the method described in [5] is large scale entity resolution in graphs.

5.1 Related Work

There is a basic set of node-level features that we work on. We started with the degree, clustering coefficient and number of components induced by the egonet when it breaks. [5] describes that simple aggregation of these basic features can yield very interesting results. Indeed, we tried sum, mean, min, and max functions for the aggregation of basic neighbors features for every target node. Sometimes features obtained this way are called *regional features*. In [5] there is even a method to generate them recursively (sum of sum of sum...) and determine the optimal level of recursion through information theoric measures.

However, such recursive feature generator is generally not trivial to implement. Nonetheless, even at the second level the results seem promising. An example is the sum of the clustering coefficient, or the min community size among neighbors that tend to increase recall.

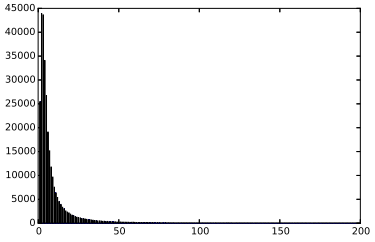
5.2 Extracting homonyms and feature interference.

Most progress was done by trying out different combination of features. However, we must mention that when doing aggregations with sum, mean and max functions heavy tailed distributions started to emerge. This is rather inconvenient when trying to separate homonyms from the rest. Such distributions have a long support, and no matter the transformations we tried the tail always interferes with homonyms. Even if it does so slightly, it represents a problem for us because of the proportions. Sometimes we though we had achieved very/ good separability by looking at the histograms, but really just the tip of the non-homonyms tail contains twice the total number of homonyms. This is why we say this is more an anomaly detection problem than classification. We have 1000 times more noisy than data (from a homonym perspective). Here we present histograms to demonstrate how we think of the dimensions of the problem. Since comparing homonym and non-homonym counts doesn't make sense, in each histogram we focus on relative frequencies.

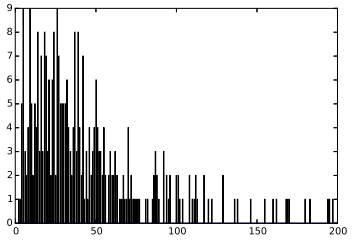
There are automatic ways to find good feature combinations, like non-linear kernels, neural networks. Nonetheless, all classifiers are really sensitive to even adding one feature. With classifiers that create these non-linear features internally it is not easy to see which features boost performance. The histograms in Fig.3 were drawn with samples of 300 000 nodes, where the homonym-non-homonym proportion is constant and equal to 0.0012. The most effective features were simple combinations, e.g. multipliying the log-degree with the conductance induced by the cut a nodes egonet from the rest of the network.

In Fig.4 one can see how extreme is the relative proportion for non-homonyms with high clustering coefficient. Nonetheless homonyms tend to have lower part of the support. This confirms our experiments in the previous section.

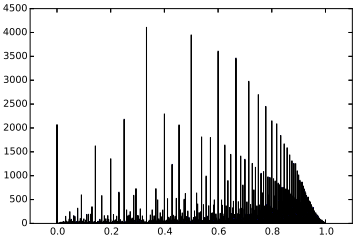
The DBLP feature has high recall because nearly all-homonym's egonets do break. However, as we mentioned, the fact that there's a peak of a 100000 non-homonyms that break as well, completely obfuscates this method's precision. Note that the clustering coefficient concentrates around 0.2, while that value is really a minimum for its counterpart. This turns out to be a good enough amount of interference for our purposes.



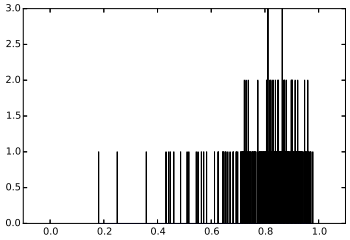
(a) Degree dist. for non-homonyms.



(b) Degree dist. for homonyms.



(c) Conductance dist. for non-homonyms.



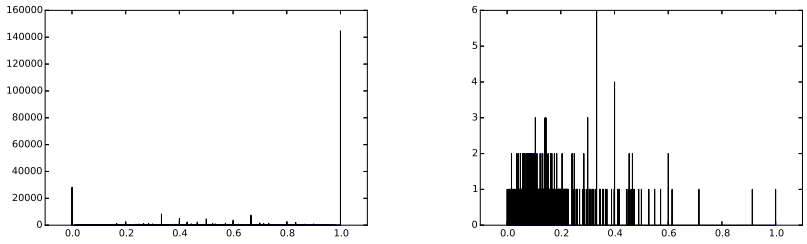
(d) Conductance dist. for homonyms.

Figure 3: For many features, homonyms lie at the end of the tail but because of unbalanced numbers of homonym and non-homonyms these are not enough to disambiguate. However conductance is high for homonyms and can be used as a multiplicative filter

5.3 Mining community structure

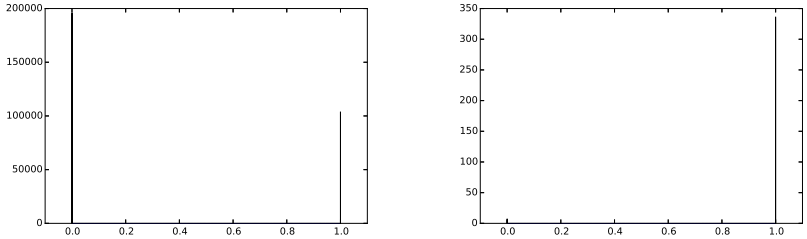
When addressing the homonyms problem it is quite natural to search valuable information in community structure. As DBLP team has pointed out, known homonyms normally join "distant groups of researchers". This is exactly the origin of their idea about looking at egonets. In contrast Natural Language Processing experts would try to find differences from context in text. Through topic modeling the distribution of topics for each author can be obtained. Then, taking for each author the top most likely topics, we could represent the entire network by these topic communities. And, again try to select homonyms by looking for entities that join distant domains.

However our approach was to exploit community structure in the graph as a whole and come up some statistics. To our surprise, we found two very successful features to mine the community structure. The first one



(a) Clcff dist. for non-homonyms. (b) Clcff dist. for homonyms.

Figure 4: Clustering coefficient distribution.



(a) DBLP feature distribution for non-homonyms. (b) DBLP feature dist. for homonyms.

Figure 5: The DBLP method as a feature.

is the number of *different* communities to which neighbors of a given node belong; the second one is a bit more elaborate and also the one that provided better results. It is a measure for similarity among communities that we call *community similarity*. To be able to calculate it, a method that finds overlapping community structure is needed. We employed the BIGCLAM method [3], that we describe now shortly.

5.4 Community detection with BIGCLAM

BIGCLAM is a third generation method for community detection. It is third generation because it adresses as a principal concern that was not forseen in previous methods: the issue of overlaps in communities (see [3] for a comprehensive discussion). BIGCLAM is based on the Affiliation Graph model (AGM), which is a family of generative models previously used in sociology

from a long time ago. In recent years very nice theoretical properties of this models were proven [4]. The affiliation graph model is in essence an overlapping $G(n, p)$ and it has really great expressive power because it can generate any network topology to some extent.
BIGCLAM fits the adjency matrix of the given graph to an AGM, the goal is to learn the latent factors which represent the communities.

With a clever formulation for the target function to optimize and a good implementation in C++ [10], this method has unprecedented performance in terms of speed, to the point that we were able to find the top 50 000 communities for the entire DBLP co-author graph running the code on a personal computer. Finding communities at such scale and tunning parameters for the community hierarchy to determine would have been incredibly more difficult with other standard methods.

5.5 Community similarity

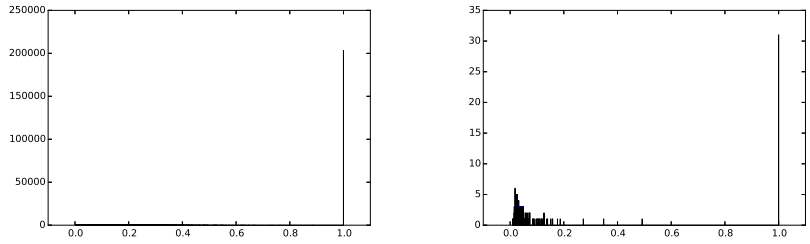
Def. Let $G = (V, E)$. The community graph $G^c = (V^c, E^c)$ defined as follows: $\forall v \in V^c, v$ corresponds to a community com_v in G.
$\forall u, v \in V^c, (u, v) \in E^c \iff$ the two corresponding communities $com_u, com_v \subseteq V$ are such that $|com_u \cap com_v| \neq \emptyset$.
We define the weighting functions $w : V^c \to \mathbb{N}$ and $e : E^c \to \mathbb{N}$:
$w(u) = |Com_u|$ where $Com_u \subseteq V$. $e(u, v) = |Com_u \cap Com_v|$ where Com_u and Com_v are the communities representing u, v respectively.

- $\forall u \in V$ we define $K(u) = \{x \in V^c : u \in Com_x\}$ the set of communities to which u belongs.
- We also denote by $G^c[K(u)] = (E^c_u, V^c_u)$ the subgraph induced by $K(u)$ in G^c.

The *community similarity* for $u \in V$ is:

$$\psi_u = \frac{2 \sum_{e_c \in E^c_u} e(e_c)}{(|V^c_u| - 1) \sum_{v_c \in V^c_u} w(v_c)}$$

ψ_u is supposed to measure how related are the communities to which u belongs to. It does so by counting the intersections these communities have. When two communities with hundreds of nodes each have very few nodes in common it is not as a significant intersection as it would be for two communities with less than ten nodes each. The easiest way to think about this measure is edge by edge in the community graph. We count twice each



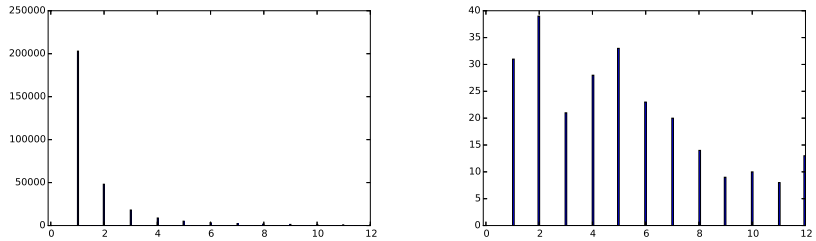
(a) ψ_u dist. for non-homonyms. (b) ψ_u dist. for homonyms.

Figure 6: Our community similarity measure.

edge in the community graph and divide by sum of the weights of the vertices this edge is incident on.

We can validate from Fig.6 our intuition about the role of homonyms connecting distant communities, so ψ concentrates on the lower part of the support.

For this sample size, there are around 350 homonyms and only 0.1 of them take value 1.0. While for non-homonyms a much greater ratio of 0.66 take value 1.0. This is because most normal vertices belong to at most two or three communities Fig.7. Our experience in disambiguating real authors confirm that this is really not far from reality. We observe from Fig.6 that there is little interference with ψ_u as we had for the clustering coefficient too. Combining the clustering coefficient and ψ_u provides the best feature we have (Section 7).



(a) Memberships dist. for non-homonyms. (b) Memberships dist. for homonyms.

Figure 7: Number of community memberships per node.

6 Procedure

The goal of this section is two fold: first it is to present the general performance of our approach, with the DBLP method, we will refer to it as *Baseline*. Then we comment the caveats when training the classifiers for the type of data we have.

In order to find a suitable evaluation criterion we first have to try with several sample sizes. This is important for this problem for two reasons. If we take very small samples, here we consider 10 000 to be small, then the amount of homonyms we obtain is about 10, which is not enough to train the classifier.

On the other hand, if we take too large samples, like 500 000, then interference in the features is more important as outliers from the non-homonyms class are more likely to appear. By successive experiments, we found 200-300 thousand to be a good sample size. We present two sample sizes in Tables 1 and 2. These are the mean scores obtained by 20 samples for each. Training and validation was done with stratified k-fold. Stratified means that we keep the homonym to non-homonym ratio in each slice. This is needed because of the extreme skewness of this distribution.

Table 1: Mean scores for sample size 300 000, and no perturbation.

	Precision	Recall	Accuracy
Baseline	0.003	0.979	0.654
Naive P.	0.332	0.023	0.998
Naive R.	0.004	0.958	0.724
Comb. P.	0.6	0.021	0.998
Comb. R.	0.003	0.958	0.708

Table 2: Mean scores for sample size 100 000, and no perturbation.			
	Precision	Recall	Accuracy
Baseline	0.003	0.99	0.65
Naive P.	0.332	0.023	0.998
Naive R.	0.004	0.958	0.725
Comb. P.	0.566	0.014	0.998
Comb. R.	0.004	0.958	0.708

We call *Naive*, the set of features composed by the degree, clustering coefficient and DBLP method classification as a feature. Then, we compare to the most stable combination, *Comb.* in the tables, that provided us with good practical results: degree, clustering, DBLP, conductance, nb. of communities in neighbors and the community similarity (ψ). The exact combination involves a few transformations for high value features like the degree or nb. of communities in neighbors. This two former quantities are multiplied by conductance as it turns out to be a good filter. We also multiply ψ with the clustering coefficient, see section 5 for some intuition. *Comb. P.* means that the classifier hyper parameters are tunned to get high precision, respectively with recall in *Comb. R.*

We tried various types of classifiers among which: svm, logistic-regression, random forest. Nonetheless, performance was quite similar across all of them despite extensive tunning. The set of features is what really matters. There is an interesting behaviour that we make explicit in Table 1 and Table 2. For all classifiers, there's always an hyper parameter for which slight variation implies either high precision or high recall (γ in svm for example), but not both. In order to fix this, we even tried bagging methods, like adaptive boosting, so that we could get an intermediate. Nevertheless, even with bagging this trade off holds true, we tried hard to tune it with no significant difference. We can explain that such trade off is more fundamental. Because of the large amount of interference within features, even with non-linear combinations there's usually this thin line between getting all homonyms or just some.

7 Results

7.1 Choosing a suitable classifier mode.

In this section we present our two main results. First is why we care more about precision. Second, we show the tendency in the precision score as DBLP is perturbed with more collisions.

So, how should we choose? For a classifier that has 0.99 recall but 0.001 precision. This means that we have a search space (for potential homonyms) 1000 fold the size of our homonym set. For a typical sample this would mean 80% of it. Even if it is some progress, for such a large dataset this remains unpractical. The DBLP method does exactly that, with results far from satisfactory. In recall mode our classifier achieves very similar performance, except in accuracy (+ 0.1 in avg.) . This can be advantageous if you use graphical models for the fine processing. It would imply 0.1 more evidence variables, which from large samples or even the entire data set is quite important.

However, the metric we have put our attention on is precision. From our practical experiments, and manual verification, it turns out that missing out most homonyms while getting good precision for a small subset gives extremely good practical results. The new homonyms found were verified by hand, with external resources such as Linkedin profiles, personal pages, list of publications and when these were not available we even sent messages to the authors. We know these are unknown to DBLP because the homonyms they know about are explicitly marked. The list of homonyms we present in the Appendix are confirmed new homonyms.

With an average precision of 0.7 in our processing batches, we get an average of 1.5 new true homonyms per sample. It took 33 samples to get the 20 homonyms we list. Each sample is of size 300 thousand, but the classifier throws 10 to 15 potential homonyms. This is why we were able to do manual disambiguation. A graphical model should do the disambiguation in no time for such small number. Considering that DBLP's algorithm has virtually no hits, and most are found by personal complaints. We consider the precision mode of our classifiers is the best way to use our method.

To end this section, we want to display a bit of the reasons we see on why this method is robust. Introducing even 1.5 % of collided vertices from the total of DBLP is simulates really a great deal of homonyms for the known ones. We do so to illustrate, how things could change in future states of this network. As we predicted with the models in section 4, the DBLP method increases precision monotonically, as it only depends on whether egonets

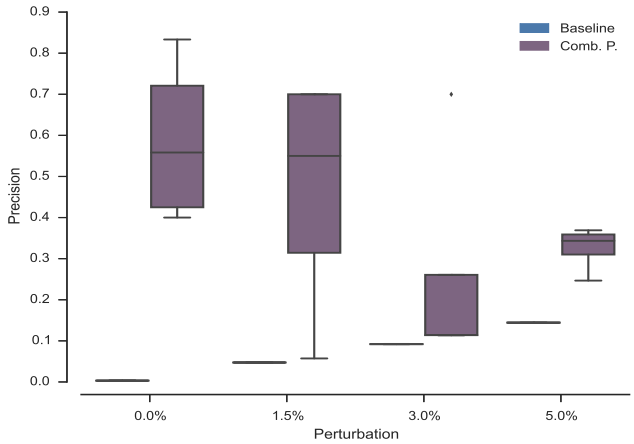


Figure 8: The evolution of precision as we introduce more collisions in the network. The precision for the DBLP-method increases monotonically with virtually no variance as we observed with the generative models in Section 4. Our method outperforms the DBLP method with no perturbation. Yet, its performance does not have a clear tendency as we increase the number of collisions. The study of these tendencies is one of the open questions we leave in this project.

break. At 0.144 it is actually a really good method, as it always keeps recall high. Nevertheless, our method in precision mode doesn't really seem to change this metric. The intuition we have about it, is that the features we learn are really scale free. And the role [6] of an homonym doesn't really change until more extreme amounts of perturbation.

8 Discussion

We now comment specifically about our experience when collecting the detected homonyms and doing disambiguation. Some of these conclusions are really intuitive, and one could think of them a priori. It is nice, however, to have such qualitative validation of the results. On the other hand some of our findings inspire some ideas for extending the method and model situations we did not foresee.

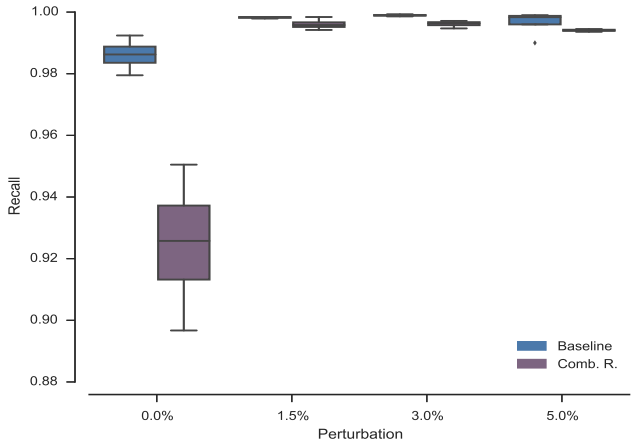


Figure 9: The evolution of recall as we introduce more collisions in the network. For small amounts of perturbation, we don't see much variation. Nonetheless, the DBLP method always has higher recall.

8.1 Applicability

Since we basically had no assumption regarding homonym formation. We believe that the homonym formation process we studied is really general and could be applicable to datasets where this is a common phenomenon. One example are news articles: since people write about a trending set of topics it is natural that titles do not vary much and collisions can happen frequently. Another example are e-commerce sites, specially those that sell products from different retailers. Since the inventory can be difficult to keep up with, and input mistakes are done all the time, we expect products with slightly different catalog name will collide in the same record.

8.2 Qualitative causes for homonym formation

In the disambiguation phase, we observed that true homonyms very likely have the same ethnicity. Also, these names are in general of common combinations of names in their respective countries. As expected, middle names abbreviations seem to be highly correlated with with homonym formation. What was not so obvious is that homonyms normally have very different ages (e.g. one of them was publishing a lot during the 80's and the 90's and its counterpart only has a few recent papers). It is also interesting to see that homonyms frequently included people that are not from the core fields of what DBLP is supposed to cover. For example, geology, chemistry,

agriculture or some of the fundamental sciences like math or physics. This is perhaps why these people do not send complaints regarding their records: they might not even know they are in DBLP.

8.3 Limitations encountered and possible extensions

Ambiguity with authors that work with subjects that are not part of the DBLP core is actually an important source of false positives for our method in precision mode. We call these type of authors *the Newcomers*. The Newcomers are often authors that authored papers where a large amount of people participated from different domains. For example, a geologist is likely to coauthor a paper about satellite communications. For our method, the problem with Newcomers is that participating in broad-domain papers, they happen to join distant communities. Furthermore, their neighborhood conductance is really high.

We believe that our approach could be further developed by studying a parallel graph, namely, the one that is used for disambiguation. In our case, this would be the co-appearance publication graph. For example, the Newcomer's issue could be solved by just looking at the degree in such parallel graph. This is because Newcomers only appear on one or two publications for the DBLP record. In addition, there is a rich community structure that can be mined. Conferences, venues, and topics should appear in the detected hierarchy. The types of similarity measures for communities can be akin the one we used. We encourage people interested on taking such approaches to develop more of them, as this is a rich and flexible source of information.

A final note with respect to the precision/recall trade-off : this is really the longer path to go. When generic detection algorithms are able to achieve about 0.9 in recall and 0.7 in precision, the homonym/synonym issue will be almost solved for many networks.

9 Appendix

9.1 A list of twenty unknown homonyms to DBLP

Many of the following homonyms listed have more people within the record that the ones we present. Our evidence for the authenticity of the following homonyms is not based on the affiliation only. We read biographies, list of publications and Linkedin pages when available. We show necessary information to put in evidence that these are different people. A quick search in the web can verify their authenticity. itemize

Alireza Javadi.

1. PhD student in EECS. ETS Montreal.
2. Mathematician. Tabriz Branck, Islamic Azad University, Tabriz, Iran.

Andrew Coyle.

1. Researcher in telecommunications. PhD in 1990. University of Adelaide, Australia.
2. Software systems. Phd in 2003.
3. Librarian. Rochester Public Library, NY USA.

Qi Li.

1. Data science. Woman. Northern Kentucky University USA.
2. Data mining, bio-informatics. Man. Western Kentucky University USA.
3. Complex networks. School of Information Science and Technology, Donghua University, Shanghai, China.

Zhiyun Yin.

1. Mathematician. Department of Information, Hunan Business College Changsha, China.
2. Signal processing. Faculty at Automation of Information Engineering, Xi'an University of Technology, China.

Di Wu.

1. EECS. Faculty at Sun Yat-Sen University, China.
2. Bioinformatics. Post-doc. University of Texas Austin, USA.

Wolfgang Kainz.

1. Cartography and geographic information.PhD at Graz University. Faculty at University of Vienna, Austria.
2. Medical imaging, radiology. PhD at University of Vienna. FDA USA.

Chang Liu.

1. Large scale data analysis. Shanghai Jiao Tong University.
2. Education, e-learning. Beijing Normal University, School of Psychology, China.

Gang Li.

1. Network science. Keylab of Net. Science and Technology, Institute of Computing Tech, China.
2. Software ware-house. Melbourne, Australia.
3. Medical imaging. Northwestern Polytech. Univ., Xi'an, China.

Vijay Kumar.

1. Computer vision, machine learning. IIT Hyderabad, India.
2. Senior researcher, concurrency, databases. University of Missouri, Kansas City, USA.
3. Robotics. Dept. of mechanical engineering. University of Pennsylvania, USA.

Qi Zhang.

1. Biostatistics and medical information. University of Wisconsin Madison, USA.
2. Cloud computing. Univ. of Waterloo, CA.
3. Complex networks. Sun Yat-Sen University, China.

Stahl, Daniel.

1. Software developer, cloud computing. Ericsson AB, Linkoping, Sweden.
2. Mathematical modeling, medical diagnosis. Lund University, Sweden.

Mark Stewart.

1. Philosophy and pharmacology. Faculty at Sunny downstate medical center, USA.
2. Scientific code development, symbolic and semantic analysis. NASA, OHIO, USA.

Tao Zhang.

1. Librarian. Purdue University Libraries, USA.
2. Image processing, vision. PhD in Japan. Faculty head of department. Tshingua, Beijing, China.

3. Telecommunications. PhD in Massachusetts. Telecordia, USA.

Ming Liu.

1. Nano Fabrication. Director of Lab. Beijing, China.
2. Machine learning, data mining. PhD student. Changchun University, China.
3. PhD student, Harbin Institute of Technology, China.

Lawrence W. Hunter.

1. Algorithms, data structures. Phd in 1971. Wisconsin, USA.
2. Materials, fibers. Phycisyst. John Hopkins University, USA.

Ning-Li.

1. Applied Mathematics. Shanghai University, China.
2. Environmental planning. Geo spatial technology. College of Env. Planning, Kaifeng, China.
3. Scandal for corruption. Univ. of Agriculture and Technology, China.
4. Formal semantics and program debugging. China Ship Research and Dev. Academy, china.

Yu Zhou

1. Language recognition. Institue of Information engineering, Beijing, China.
2. Imaging, databases. Nanjing University, China.
3. Pure math. Tsinghua, Beijing, China.

Hui Liu. DBLP detected one homonym author inside this record, yet, we found others undetected.

1. Biophysics. Wuhan, China.
2. Communications and signal processing. Men. PhD at Texas Austin. Professor at University of Washington.
3. Computer and communication networks. Woman. PhD at Georgia State University. Missouri State University, USA.
4. Multi agent-systems, complex systems. Associate Professor, Huazhong University of Science and Technology, China.

Xiadong Wang.

1. Currently PhD student. in EECS. Ohio State University, USA.

2. Electrical engineering. Professor at Columbia University, USA.
itemize

References

[1] Michael Ley. *DBLP-Some Lessons Learned, 2009.*

[2] Dan J. Wang, Xiaolin Shi, Daniel A. McFarland, Jure Leskovec. *Measurement error in network data: A re-classification, 2012.*

[3] Jaewon Yang, Jure Leskovec. *Overlapping Community Detection at Scale: A Nonnegative Matrix Factoriaztion Approach, 2012.*

[4] Silvio Lattanzi, D. Sivakumar. *Affiliation Networks, 2009.*

[5] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, Christos Faloutsos. *It's Who You Know: Graph Mining Using Recursive Structural Features, 2011.*

[6] Keith Henderson, Brian Gallagher, Lei Li, Sugato Baso, Danai Koutra, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, Christos Faloutsos. *RolX: Structural Role Extraction and Mining in Large Graphs, 2012.*

[7] Hsin-Tsung Peng, Cheng-Yu Lu, William Hsu, Jan-Ming Ho. *Disambiguating authors in citations on the web and authorship correlations, 2012.*

[8] I. Bhattacharya, L. Getoor. *Entity Resolution in Graphs, 2005.*

[9] L.Getoor, A. Machanavajjhala. *Entity Resolution: Theory, Practice and Open Challenges, 2012.*

[10] Jure Leskovec and the SNAP community. *Social Network Analysis Project. https://github.com/snap-stanford/snap*

[11] Michael Ley.
http://dblp.uni-trier.de/faq/How+does+dblp+handle+homonyms+and+synonyms

[12] *MLCONF 2014 SAN FRANCISCO*

[13] The DBLP team. *http://dblp.uni-trier.de/*