Algorithms for Database Systems (Seminar)

Collective Graph Identification

Floran Gmehlin
fgmehlin@student.ethz.ch
# Table of content

**Algorithms for Database Systems (Seminar)** ................................................................. 1  
**Table of content** ........................................................................................................... 2  
**Introduction** ................................................................................................................. 1  
**Problem Statement** ..................................................................................................... 2  
**Coupled Collective Classifiers** .................................................................................. 4  
- Basics ............................................................................................................................ 4  
- Observed and predicted variables ................................................................................ 4  
- Features .......................................................................................................................... 5  
- Feature Weights ............................................................................................................. 6  
- Conditional Markov Network ...................................................................................... 7  
- Probability Distribution .............................................................................................. 7  
- Graph construction ...................................................................................................... 8  
**Conclusion** .................................................................................................................. 8
**Introduction**

While the amount of data produced, transmitted and processed grows larger over time, we become more and more interested in analyzing networks that describe such interactions. Whether we talk about communication networks, collaboration network or transmitted diseases networks, it is in fact not trivial to illustrate the true form of an underlying graph. Take any company, IBM for example. Every employee communicates internally via email with other employees. Drawing a graph about these communications would give hints about who interacts with whom, but hardly give information about the hierarchical structure of the firm. There is a clear distinction between observed data and the true form of the hidden sociological or technological network. We can further illustrate this problem with scientific papers and their references. Predicting whether a paper cites another one, allows determining whether two papers cite common papers and thus whether they have the same topic, which could give rise to the question of whether they are the same paper or not.

The publication summarized in this document aims to introduce a new approach of discovering such graphs, called C³ (Couple Collective Classifiers). However, this document will focus on the global functioning of the method without exposing details about the experimentation itself and its results.

Problem Statement

Based on observed networks, C³ introduces the problem of graph identification, which intents to discover the real structure of their concealed graph. The problem is cast as a probabilistic inference task, in which we must infer the nodes, edges and node labels of a hidden graph, based on evidence provided by the observed, raw graph.

Three different tasks have to be considered in order to achieve such results: entity resolution, link prediction and node labeling. While each of these problems has been studied separately, C³ is the first approach to consider them together as a coherent task. Figure (a), (b) and (c) illustrate a simple example of an email communication network within a firm.

**Entity Resolution:** Some nodes of the input graph may refer to the same person. The process of entity resolution aims to recognize and merge them. In Figure (a), we notice that nsmith@msn.com and neil@example.com potentially belong to a certain “Neil Smith”.

**Link Prediction:** This process aims to discover new edges in the output graph based on which information the input graph actually shows. For example, we may consider a hierarchical relationship between Anne Cole and Mary Taylor in Figure (b).

**Node Labeling:** Node labeling allows, in our example, to find out which role each person has in the company.
Each of the three tasks (entity resolution, link prediction, node labeling) informs the other. The goal of $C^3$ is to solve them simultaneously, using an iterative procedure, and thus allowing observed information and results to scatter among them to get better results.


**Coupled Collective Classifiers**

**Basics**

First of all, $C^3$ takes as input a graph $G(V, E)$ with $V$ being the set of vertices and $E$ the set of directed edges. Each node $v \in V$ describes an entity (or a reference to it) as of each edge $e \in E, e = (v_i, v_j)$ represent an interaction between nodes $v_i$ and $v_j$. In the previous example, we can think of nodes being people and edges being email sent between them. (Figure (a)).

Secondly, nodes and edges can have attributes associated to. An example of attributes associated to a node $v_i$ would be the words that appear in the email written by person $i (A_i)$, and the number of emails that have been sent from person $i$ to person $j$ for the associated edge $(A_{ij})$. Thus, the attributes of all input nodes and edges is denoted by $A = \{A_i\} \cup \{A_{ij}\}$.

**Entity Resolution**

As stated above, entity resolution aims to recognize co-referent entities. $C^3$ uses binary random variables $R = \{R_{ij}\}$ with $i, j = 1, \ldots, |V|$ to indicate whether nodes $v_i$ and $v_j$ are co-referent.

**Link Prediction**

$C^3$ also uses binary random variables $L = \{L_{ij}\}$ with $i, j = 1, \ldots, |V|$ to indicate whether there will be a link between nodes $v_i$ and $v_j$ in the output graph.

**Node Labeling**

As a node can take many possible values as label, $C^3$ uses a random variable for each node as following: $N = \{N_i\}$, with $N_i \in \{1, \ldots, k\}, i = 1, \ldots, |V|$

**Observed and predicted variables**

Each set of variables is separated into two groups: the observed variables (evidences) from the input graph and the predicted ones (targets).

Observed variables: $R_o, L_o, N_o$

Predicted variables: $R_p, L_p, N_p$

Note that attributes $A$ and edges $E$ are both assumed to be observed as well. Hence, evidence is defined as following: $X = R_o \cup L_o \cup N_o \cup A \cup E$ and target: $Y = R_p \cup L_p \cup N_p$
Features

Feature in C³ allow capturing dependencies in a local and relational way. Local features helps to determine dependencies between a single observed and predicted variable whereas relational features focus on the interaction between multiple predicted variables.

An example of a local feature could be the String similarity of two different observed email addresses belonging to the same person: fgmehlin@student.ethz.ch and floran.gmehlin@gmail.com.

Relational features are further decomposed into two categories:

- **Intra-relational features:** Centered in a single task (ER, LP or NL), these features help broadcasting information among its variables.
  - Consider for example a node $N_i$ and its observed (from the input graph) neighbors for which we have predicted their node labels. An intra-relational feature on node $N_i$ would characterize the condition that it depends on the predicted labels of its neighbors.
  - Another interesting example in the case of link prediction would be to take two scientific papers and analyze their content words. Two papers are likely to share a link based on the similarity of their content. For each word $w_i$, $\text{HasWord}(X, w_i) \land \text{HasWord}(Y, w_i)$ would give a useful hint whether paper X and paper Y should share a link or not.

- **Inter-relational features:** These kinds of features help propagating information between each of the three tasks.
  - Consider this time a node $N_i$ for which we have predicted a few neighbors. In terms of node labeling, an inter-relational feature would represent the condition that the label of node $N_i$ depends on its inferred neighbors along the predicted edges on the output graph.
  - Another interesting example would be to consider two programmers inside a company who do not know each other, but work on the same project. They both report via email to their (common) manager. Let us assume that we predicted a link between them (the programmers) after having analyzed the similar content of their emails. An inter-relational feature would capture the dependence of their node label (programmer) based on the predicted link.
Feature Weights

With every feature is associated a weight. When computing the target variables \( R_p, L_p \) and \( N_p \), we are interested in knowing how strong the impact of a feature will be. Note that a feature can be defined over more than one variable, (e.g. \( f(R_p, L_p, x) \)) and therefore should normally have the same weight. In \( C^3 \), a simplification is made in the sense that such a feature would be weighted separately, therefore allowing the weight-learning algorithm to find the optimal value for each of the variables.

**Algorithm 1  \( C^3 \) Semi-supervised Weight Learning**

| input: \( f^{local} \), a set of local features                        |
| \( f^{relational} \), a set of relational features                  |
| \( y^{observed} \), values of observed variables                   |
| \( Y^{predicted} \), predicted variables                           |
| \( x \), evidence variables                                        |
| output: \( w \), weights of \( f^{local} \cup f^{relational} \)      |
| \( w^{local} \), weights of \( f^{local} \)                         |
| calls: \( LearnWeights(f, y, x, C) \), which returns weights of features \( f \) given observed variables \( y \), evidence \( x \) and classifier \( C \) |
| \( InferValue(Y, f, w, x, C) \), which returns the MAP value of variable \( Y \) given features \( f \), their weights \( w \), evidence \( x \) and classifier \( C \) |

1: \( w^{local} \leftarrow LearnWeights(f^{local}, y^{observed}, x, SVM) \)
2: for each \( Y \in Y^{predicted} \)
3: \( y^{predicted} \leftarrow InferValue(Y, f^{local}, w^{local}, x, SVM) \)
4: \( f \leftarrow f^{local} \cup f^{relational} \)
5: \( w \leftarrow LearnWeights(f, y^{observed}, x \cup y^{predicted}, SVM) \)
6: return \( (w, w^{local}) \)

The weight-learning algorithm makes use of multi-class support vector machines\(^1\) that, in a first time, uses only local features to learn the most appropriate local weights in order to infer target values. This is actually done because the set of observed data is incomplete and thus would makes it difficult to compute the values of relational features. Each task has its proper SVN for computing their respective target values.

---

\(^1\) A multi-class support vector machine is preferred as node-labeling variables \( N \) can be assigned to one of more than two possible values. Recall from section « Basics » that a node can take more than two possibilities values.
Once local weights have been learnt, the algorithm can obtain a map value of target variables \( Y \) given their local features and associated weights in order to later compute the weights of relational features.

**Conditional Markov Network**

In \( C^3 \), the input graph that will be processed is considered to be a Markov Network.

A Markov network is an undirected graph where nodes (random variables) satisfy a Markov property. In our case, as we have defined what constitute an evidence i.e. observed data, therefore a Markov property would be: if node \( E \) is an evidence, then we can say that \( C \) is conditionally independent of the rest of the graph, given \( E \).

**Probability Distribution**

We would like to know which values our random variables are likely to take, given the initial evidence, or observed data.

The Hammersley-Clifford Theorem states that a probability distribution with a positive density, w.r.t an undirected graph \( G \) satisfies one of the Markov properties if and only if it can be factorized over the cliques of the graph \( G \).

\[
(1) \quad P(y|x) = \frac{1}{Z(x)} \prod_{c \in \mathcal{C}} \Phi_c(x_c, y_c)
\]

\( \Phi_c \) being the associated potential of a clique \( c \) of the graph: a non-negative real-valued function representing the state of the clique \( c \). In this case, features are used to capture the state of a clique, that is, determine the dependency between the vertices (random variables) of a clique.

\[
(2) \quad P(y|x) = \frac{1}{Z(x)} \exp \left( \sum_{c \in \mathcal{C}} w_c \cdot f_c(x_c, y_c) \right)
\]

with \( Z(x) = \sum_{y} \prod_{c \in \mathcal{C}} \Phi_c(x_c, y_c) \)

Evaluating \( Z(x) \) requires that we sum over all the possible assignments of all the values \( y \), which is intractable since it is exponential in \( |Y| \).
A pseudolikelihood approximation is used in order to reduce the complexity. This method suggests to use a Markovian approach of the problem, i.e. specify the conditional probability of each value $y_i$, knowing all the other $y_j, i \neq j$ values. This breaks down the problem into smaller sub problems solvable in linear time over $|Y|$ to evaluate $Z(x)$.

\[(3) \quad P^*(y|x) = \prod_i P(y_i|y-i,x) \text{ with } y-i = y_i, \ldots, y_{i-1}, y_{i+1}, \ldots, y_m\]

With help of this pseudolikelihood and the definition of the variables made previously, we can finally reach the joint probability distribution used in $C^3$:

$$P^*(r_p, l_p, n_p|x) = \prod_{y \in r_p \cup l_p \cup n_p} \frac{\exp(\sum_{f \in F, y \in y_f} w_f \cdot f(x_f, y_f))}{Z(y \in y_f, x)}$$

**Graph construction**

In the end, the output graph can finally be constructed using the result set of predicted variables for each of the three tasks (entity resolution, link prediction and node labeling) in their respective order. An entity node is the output graph is created for each collection of co-referent references (e.g. if nsmith@msn.com and neil@example.com are shown as co-referent, a single entity will be implied). Entity linkage is accomplished based on the indication shown by their corresponding variables $L$. As for node labeling, the values of the variables $N$ corresponding to their references will suggest a suitable node label.

Errors may however still occur. Inconsistent assignment of variables is possible and should be dealt with prior to generating the output graph by enforcing properties, adding edges or changing node labels.

**Conclusion**

While $C^3$ is a novel approach, it shows outstanding results that significantly outperform all the other methods with the same goal. The wide variety of features used by $C^3$ has proven to be directly correlated with its performance. Graph identification is indeed an emerging problem nowadays as we live in an era where the amount of information created and transmitted tends to grow exponentially. Today, around 200 millions emails are sent over the network every minute. While these observed data are likely to be noisy, it is important to use methods capable of successfully and efficiently processing them.