Information Integration of Partially Labeled Data

Steffen Rendle and Lars Schmidt-Thieme

Information Systems and Machine Learning Lab, University of Hildesheim
srendle@ismll.uni-hildesheim.de, schmidt-thieme@ismll.uni-hildesheim.de

Abstract. A central task when integrating data from different sources is to detect identical items. For example, price comparison websites have to identify offers for identical products. This task is known, among others, as record linkage, object identification, or duplicate detection.

In this work, we examine problem settings where some relations between items are given in advance – for example by EAN article codes in an e-commerce scenario or by manually labeled parts. To represent and solve these problems we bring in ideas of semi-supervised and constrained clustering in terms of pairwise must-link and cannot-link constraints. We show that extending object identification by pairwise constraints results in an expressive framework that subsumes many variants of the integration problem like traditional object identification, matching, iterative problems or an active learning setting.

For solving these integration tasks, we propose an extension to current object identification models that assures consistent solutions to problems with constraints. Our evaluation shows that additionally taking the labeled data into account dramatically increases the quality of state-of-the-art object identification systems.

1 Introduction

When information collected from many sources should be integrated, different objects may refer to the same underlying entity. Object identification aims at identifying such equivalent objects. A typical scenario is a price comparison system where offers from different shops are collected and identical products have to be found. Decisions about identities are based on noisy attributes like product names or brands. Moreover, often some parts of the data provide some kind of label that can additionally be used. For example some offers might be labeled by a European Article Number (EAN) or an International Standard Book Number (ISBN). In this work we investigate problem settings where such information is provided on some parts of the data. We will present three different kinds of knowledge that restricts the set of consistent solutions. For solving these constrained object identification problems we extend the generic

object identification model by a collective decision model that is guided by both constraints and similarities.

2 Related Work

Object identification (e.g. Neiling 2005) is also known as record linkage (e.g. Winkler 1999) and duplicate detection (e.g. Bilenko and Mooney 2003). State-of-the-art methods use an adaptive approach and learn a similarity measure that is used for predicting the equivalence relation (e.g. Cohen and Richman 2002). In contrast, our approach also takes labels in terms of constraints into account.

Using pairwise constraints for guiding decisions is studied in the community of semi-supervised or constrained clustering – e.g. Basu et al. (2004). However, the problem setting in object identification differs from this scenario because in semi-supervised clustering typically a small number of classes is considered and often it is assumed that the number of classes is known in advance. Moreover, semi-supervised clustering does not use expensive pairwise models that are common in object identification.

3 Four problem classes

In the classical object identification problem $\mathcal{C}_{classic}$ a set of objects X should be grouped into equivalence classes E_X. In an adaptive setting, a second set Y of objects is available where the perfect equivalence relation E_Y is known. It is assumed that X and Y are disjoint and share no classes – i.e. $E_X \cap E_Y = \emptyset$.

In real world problems often there is no such clear separation between labeled and unlabeled data. Instead only the objects of some subset Y of X are labeled. We call this problem setting the iterative problem $\mathcal{C}_{iter}$ where (X, Y, E_Y) is given with $X \supseteq Y$ and $Y^2 \supseteq E_Y$. Obviously, consistent solutions E_X have to satisfy $E_X \cap Y^2 = E_Y$. Examples of applications for iterative problems are the integration of offers from different sources where some offers are labeled by a unique identifier like an EAN or ISBN, and iterative integration tasks where an already integrated set of objects is extended by new objects.

The third problem setting deals with integrating data from n sources, where each source is assumed to contain no duplicates at all. This is called the class of matching problems $\mathcal{C}_{match}$. Here the problem is given by $\mathcal{X} = \{X_1, \ldots, X_n\}$ with $X_i \cap X_j = \emptyset$ and the set of consistent equivalence relations $\mathcal{E}$ is restricted to relations E on X with $E \cap X_i^2 = \{(x, x) | x \in X_i\}$. Traditional record linkage often deals with matching problems of two data sets ($n = 2$).

At last, there is the class of pairwise constrained problems $\mathcal{C}_{constr}$. Here each problem is defined by (X, R_{ml}, R_{cl}) where the set of objects X is constrained by a must-link R_{ml} and a cannot-link relation R_{cl}. Consistent solutions are restricted to equivalence releations E with $E \cap R_{cl} = \emptyset$ and $E \supseteq R_{ml}$.

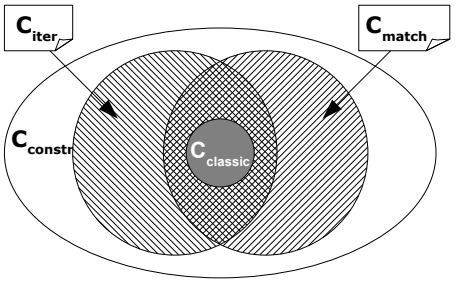

Fig. 1. Relations between problem classes: $\mathcal{C}_{classic} \subset \mathcal{C}_{iter} \subset \mathcal{C}_{constr}$ and $\mathcal{C}_{classic} \subset \mathcal{C}_{match} \subset \mathcal{C}_{constr}$.

Obviously, R_{cl} is symmetric and irreflexive whereas R_{ml} has to be an equivalence relation. In all, pairwise constrained problems differ from iterative problems by labeling relations instead of labeling objects. The constrained problem class can better describe local informations like two offers are the same/ different. Such information can for example be provided by a human expert in an active learning setting.

We will show, that the presented problem classes form a hierarchy $\mathcal{C}_{classic} \subset \mathcal{C}_{iter} \subset \mathcal{C}_{constr}$ and that $\mathcal{C}_{classic} \subset \mathcal{C}_{match} \subset \mathcal{C}_{constr}$ but neither $\mathcal{C}_{match} \subseteq \mathcal{C}_{iter}$ nor $\mathcal{C}_{iter} \subseteq \mathcal{C}_{match}$ (see Figure 1). First of all, it is easy to see that $\mathcal{C}_{classic} \subseteq \mathcal{C}_{iter}$ because any problem $X \in \mathcal{C}_{classic}$ corresponds to an iterative problem without labeled data ($Y = \emptyset$). Also $\mathcal{C}_{classic} \subseteq \mathcal{C}_{match}$ because an arbitrary problem $X \in \mathcal{C}_{classic}$ can be transformed to a matching problem by considering each object as its own dataset: $X_1 = \{x_1\}, \ldots, X_n = \{x_n\}$. On the other hand, $\mathcal{C}_{iter} \not\subseteq \mathcal{C}_{classic}$ and $\mathcal{C}_{match} \not\subseteq \mathcal{C}_{classic}$, because $\mathcal{C}_{classic}$ is not able to formulate any restriction on the set of possible solutions $\mathcal{E}$ as the other classes can do. This shows that:

$$\mathcal{C}_{classic} \subset \mathcal{C}_{match}, \quad \mathcal{C}_{classic} \subset \mathcal{C}_{iter} \tag{1}$$

Next we will show that $\mathcal{C}_{iter} \subset \mathcal{C}_{constr}$. First of all, any iterative problem (X, Y, E_Y) can be transformed to a constrained problem (X, R_{ml}, R_{cl}) by setting $R_{ml} \leftarrow \{(y_1, y_2) | y_1 \equiv_{E_Y} y_2\}$ and $R_{cl} \leftarrow \{(y_1, y_2) | y_1 \not\equiv_{E_Y} y_2\}$. On the other hand, there are problems $(X, R_{ml}, R_{cl}) \in \mathcal{C}_{constr}$ that cannot be expressed as an iterative problem, e.g.:

$$X = \{x_1, x_2, x_3, x_4\}, \quad R_{ml} = \{(x_1, x_2), (x_3, x_4)\}, \quad R_{cl} = \emptyset$$

If one tries to express this as an iterative problem, one would assign to the pair (x_1, x_2) the label l_1 and to (x_3, x_4) the label l_2. But one has to decide whether or not $l_1 = l_2$. If $l_1 = l_2$, then the corresponding constrained problem would include the constraint $(x_2, x_3) \in R_{ml}$, which differs from the original problem. Otherwise, if $l_1 \neq l_2$, this would imply $(x_2, x_3) \in R_{cl}$, which again is a different problem. Therefore:

$$\mathcal{C}_{iter} \subset \mathcal{C}_{constr} \tag{2}$$

Furthermore, $\mathcal{C}_{match} \subseteq \mathcal{C}_{constr}$ because any matching problem $X_1, \ldots, X_n$ can be expressed as a constrained problem with:

$$X = \bigcup_{i=1}^{n} X_i, \quad R_{cl} = \{(x,y) | x,y \in X_i \wedge x \neq y\}, \quad R_{ml} = \emptyset$$

There are constrained problems that cannot be translated into a matching problem. E.g.:

$$X = \{x_1, x_2, x_3\}, \quad R_{ml} = \{(x_1, x_2)\}, \quad R_{cl} = \emptyset$$

Thus:

$$\mathcal{C}_{match} \subset \mathcal{C}_{constr} \tag{3}$$

At last, there are iterative problems that cannot be expressed as matching problems, e.g.:

$$X = \{x_1, x_2, x_3\}, \quad Y = \{x_1, x_2\}, \quad x_1 \equiv_{E_Y} x_2$$

And there are matching problems that have no corresponding iterative problem, e.g.:

$$X_1 = \{x_1, x_2\}, \quad X_2 = \{y_1, y_2\}$$

Therefore:

$$\mathcal{C}_{match} \not\subseteq \mathcal{C}_{iter}, \quad \mathcal{C}_{iter} \not\subseteq \mathcal{C}_{match} \tag{4}$$

In all we have shown that $\mathcal{C}_{constr}$ is the most expressive class and subsumes all the other classes.

4 Method

Object Identification is generally done by three core components (Rendle and Schmidt-Thieme (2006)):

1. *Pairwise Feature Extraction* with a function $f : X^2 \to \mathbb{R}^n$.
2. *Probabilistic Pairwise Decision Model* specifying probabilities for equivalences $P[x \equiv y]$.
3. *Collective Decision Model* generating an equivalence relation E over X.

The task of feature extraction is to generate a feature vector from the attribute descriptions of any two objects. Mostly, heuristic similarity functions like TFIDF-Cosine-Similarity or Levenshtein distance are used. The probabilistic pairwise decision model combines several of these heuristic functions to a single domain specific similarity function (see Table 1). For this model probabilistic classifiers like SVMs, decision trees, logic regression, etc. can be used. By combining many heuristic functions over several attributes, no

Table 1. Example of feature extraction and prediction of pairwise equivalence $P[x_i \equiv x_j]$ for three digital cameras.

Object	Brand	Product Name	Price
x_1	Hewlett Packard	Photosmart 435 Digital Camera	118.99
x_2	HP	HP Photosmart 435 16MB memory	110.00
x_3	Canon	Canon EOS 300D black 18-55 Camera	786.00

Object Pair	TFIDF-Cos. Sim. (Product Name)	FirstNumberEqual (Product Name)	Rel. Difference (Price)	Feature Vector	$P[x_i \equiv x_j]$
(x_1, x_2)	0.6	1	0.076	(0.6, 1, 0.076)	0.8
(x_1, x_3)	0.1	0	0.849	(0.1, 0, 0.849)	0.2
(x_2, x_3)	0.0	0	0.860	(0.0, 0, 0.860)	0.1

time-consuming function selection and fine-tuning has to be performed by a domain-expert. Instead, the model automatically learns which similarity function is important for a specific problem. Cohen and Richman (2002) as well as Bilenko and Mooney (2003) have shown that this approach is successful. The collective decision model generates an equivalence relation over X by using $sim(x, y) := P[x \equiv y]$ as learned similarity measure. Often, clustering is used for this task (e.g. Cohen and Richman (2002)).

4.1 Collective decision model with constraints

The constrained problem easily fits into the generic model above by extending the collective decision model by constraints. As this stage might be solved by clustering algorithms in the classical problem, we propose to solve the constrained problem by a constraint-based clustering algorithm. To enforce the constraint satisfaction we suggest a constrained hierarchical agglomerative clustering (HAC) algorithm. Instead of a dendrogram the algorithm builds a partition where each cluster should contain equivalent objects. Because in an object identification task the number of equivalence classes is almost never known, we suggest model selection by a (learned) threshold θ on the similarity of two clusters in order to stop the merging process. A simplified representation of our constrained HAC algorithm is shown in Algorithm 1. The algorithm initially creates a new cluster for each object (line 2) and afterwards merges clusters that contain objects constrained by a mustlink (line 3-7). Then the most similar clusters, that are not constrained by a cannotlink, are merged until the threshold θ is reached.

From a theoretical point of view this task might be solved by an arbitrary, probabilistic HAC algorithm using a special initialization of the similarity matrix and minor changes in the update step of the matrix. For satisfaction of the constraints R_{ml} and R_{cl}, one initializes the similarity matrix for $X = \{x_1, \ldots, x_n\}$ in the following way:

$$A^0_{j,k} = \begin{cases} +\infty, & \text{if } (x_j, x_k) \in R_{ml} \\ -\infty, & \text{if } (x_j, x_k) \in R_{cl} \\ P[x_j \equiv x_k] & \text{otherwise} \end{cases}$$

As usual, in each iteration the two clusters with the highest similarity are merged. After merging cluster c_l with c_m the dimension of the square matrix A reduces by one – both in columns and rows. For ensuring constraint satisfaction, the similarities between $c_l \cup c_m$ to all the other clusters have to be recomputed:

$$A^{t+1}_{n,i} = \begin{cases} +\infty, & \text{if } A^t_{l,i} = +\infty \vee A^t_{m,i} = +\infty \\ -\infty, & \text{if } A^t_{l,i} = -\infty \vee A^t_{m,i} = -\infty \\ sim(c_l \cup c_m, c_i) & \text{otherwise} \end{cases}$$

For calculating the similarity *sim* between clusters, standard linkage techniques like single-, complete- or average-linkage can be used.

Algorithm 1 Constrained HAC Algorithm

procedure ClusterHAC(X, R_{ml}, R_{cl})
  $P \leftarrow \{\{x\} | x \in X\}$

  for all $(x, y) \in R_{ml}$ **do**
    $c_1 \leftarrow c$ *where* $c \in P \wedge x \in c$
    $c_2 \leftarrow c$ *where* $c \in P \wedge y \in c$
    $P \leftarrow (P \setminus \{c_1, c_2\}) \cup \{c_1 \cup c_2\}$
  end for

  repeat
    $(c_1, c_2) \leftarrow \underset{c_1, c_2 \in P \wedge (c_1 \times c_2) \cap R_{cl} = \emptyset}{\operatorname{argmax}} sim(c_1, c_2)$
    if $sim(c_1, c_2) \geq \theta$ **then**
      $P \leftarrow (P \setminus \{c_1, c_2\}) \cup \{c_1 \cup c_2\}$
    end if
  until $sim(c_1, c_2) < \theta$

  return P
end procedure

4.2 Algorithmic Optimizations

Real-world object identification problems often have a huge number of objects. An implementation of the proposed constrained HAC algorithm has to consider several optimization aspects. First of all, the cluster similarities should be computed by dynamic programming. So the similarities between

clusters have to be collected just once and afterward can be inferred by the similarities, that are already given in the similarity-matrix:

$$sim_{sl}(c_1 \cup c_2, c_3) = \max\{sim_{sl}(c_1, c_3), sim_{sl}(c_2, c_3)\} \qquad \textit{single-linkage}$$

$$sim_{cl}(c_1 \cup c_2, c_3) = \min\{sim_{cl}(c_1, c_3), sim_{cl}(c_2, c_3)\} \qquad \textit{complete-linkage}$$

$$sim_{al}(c_1 \cup c_2, c_3) = \frac{|c_1| \cdot sim_{al}(c_1, c_3) + |c_2| \cdot sim_{al}(c_2, c_3)}{|c_1| + |c_2|} \qquad \textit{average-linkage}$$

Second, a blocker should reduce the number of pairs that have to be taken into account for merging. Blockers like the canopy blocker (McCallum et al. (2000)) reduce the amount of pairs very efficiently, so even large data sets can be handled. At last, pruning should be applied to eliminate cluster pairs with similarity below θ_{prune}. These optimizations can be implemented by storing a list of cluster-distance-pairs which is initialized with the pruned candidate pairs of the blocker.

5 Evaluation

In our evaluation study we examine if additionally guiding the collective decision model by constraints improves the quality. Therefore we compare constrained and unconstrained versions of the same object identification model on different data sets. As data sets we use the bibliographic Cora dataset that is provided by McCallum et al. (2000) and is widely used for evaluating object identification models (e.g. Cohen et al. (2002) and Bilenko et al. (2003)), and two product data sets of a price comparison system.

We set up an iterative problem by labeling N% of the objects with their true class label. For feature extraction of the Cora model we use TFIDF-Cosine-Similarity, Levenshtein distance and Jaccard distance for every attribute. The model for the product datasets uses TFIDF-Cosine-Similarity, the difference between prices and some domain-specific comparison functions. The pairwise decision model is chosen to be a Support Vector Machine. In the collective decision model we run our constrained HAC algorithm against an unconstrained ('classic') one. In each case, we run three different linkage methods: single-, complete- and average-linkage. We report the average F-Measure quality of four runs for each of the linkage techniques and for constrained and unconstrained clustering. The F-Measure quality is taken on all pairs that are unknown in advance – i.e. pairs that do not link two labeled objects.

$$\textit{F-Measure} = \frac{2 \cdot \textit{Recall} \cdot \textit{Precision}}{\textit{Recall} + \textit{Precision}}$$

$$\textit{Recall} = \frac{TP}{TP + FN}, \quad \textit{Precision} = \frac{TP}{TP + FP}$$

Table 2 shows the results of the first experiment where $N = 25\%$ of the objects for Cora and $N = 50\%$ for the product datasets provide labels. As one

Table 2. Comparison of F-Measure quality of a constrained to a classical method with different linkage techniques. For each data set and each method the best linkage technique is marked bold.

Data Set	Method	Single Linkage	Complete Linkage	Average Linkage
Cora	classic/constrained	0.70/0.92	0.74/0.71	**0.89**/**0.93**
DVD player	classic/constrained	**0.87**/0.94	0.79/0.73	0.86/**0.95**
Camera	classic/constrained	0.65/**0.86**	0.60/0.45	**0.67**/0.81

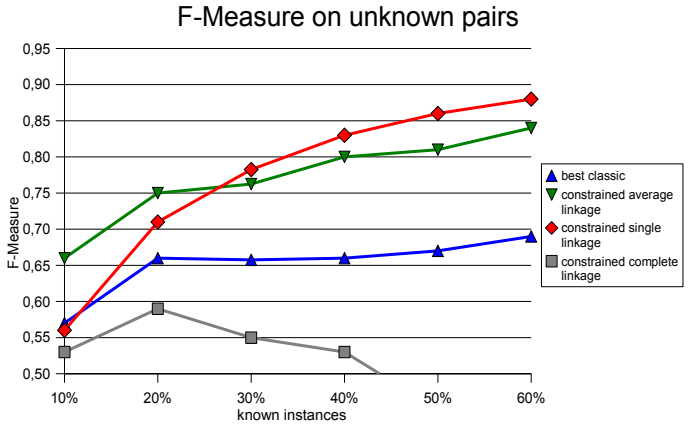

Fig. 2. F-Measure on Camera dataset for varying proportions of labeled objects.

can see, the best constrained method always clearly outperforms the best classical method. When switching from the best classical to the best constrained method, the relative error reduces by 36% for Cora, 62% for DVD-Player and 58% for Camera. An informal significance test shows that in this experiment the best constrained method is better than the best classic one.

In a second experiment (see Figure 2) we increased the amount of labeled data from $N = 10\%$ to $N = 60\%$ and report results for the Camera dataset for the best classical method and the three constrained linkage techniques. The figure shows that the best classical method does not improve much beyond more than 20% labeled data. In contrast, when using the constrained single- or average-linkage technique the quality on non-labeled parts improves always with more labeled data. When few constraints are available average-linkage tends to be better than single-linkage whereas single-linkage is superior in the case of many constraints. The reason are the cannot-links that prevent single-linkage from merging false pairs. The bad performance of constrained complete-linkage can be explained by must-link constraints that might result in diverse clusters (Algorithm 1, line 3-7). For any diverse cluster, complete-linkage can not find any cluster with similarity greater than θ and so after the initial step, diverse clusters are not merged any more (Algorithm 1, line 8-13).

6 Conclusion

We have formulated three problem classes that encode knowledge and restrict the space of consistent solutions. For solving problems of the most expressive class $\mathcal{C}_{constr}$, that subsumes all the other classes, we have proposed a constrained object identification model. Therefore the generic object identification model was extended in the collective decision stage to ensure constraint satisfaction. We proposed a HAC algorithm with different linkage techniques that is guided by both a learned similarity measure and constraints. Our evaluation has shown, that this method with single- or average-linkage is effective and using constraints in the collective stage clearly outperforms non-constrained state-of-the-art methods.

References

BASU, S. and BILENKO, M. and MOONEY, R. J. (2004): A Probabilistic Framework for Semi-Supervised Clustering. In: *Proceedings of the 10th International Conference on Knowledge Discovery and Data Mining (KDD-2004)*.

BILENKO, M. and MOONEY, R. J. (2003): Adaptive Duplicate Detection Using Learnable String Similarity Measures. In: *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining (KDD-2004)*.

COHEN, W. W. and RICHMAN, J. (2002): Learning to Match and Cluster Large High-Dimensional Data Sets for Data Integration. In: *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining (KDD-2002)*.

MCCALLUM, A. K., NIGAM K. and UNGAR L. (2000): Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching. In: *Proceedings of the 6th International Conference On Knowledge Discovery and Data Mining (KDD-2000)*.

NEILING, M. (2005): Identification of Real-World Objects in Multiple Databases. In: *Proceedings of GfKl Conference 2005*.

RENDLE, S. and SCHMIDT-THIEME, L. (2006): Object Identification with Constraints. In: *Proceedings of 6th IEEE International Conference on Data Mining (ICDM-2006)*.

WINKLER W. E. (1999): *The State of Record Linkage and Current Research Problems.* Technical report, Statistical Research Division, U.S. Census Bureau.