Noname manuscript No.
(will be inserted by the editor)

Collective Regression for Handling Autocorrelation of Network Data in a Transductive Setting

Corrado Loglisci · Annalisa Appice · Donato Malerba

Received: date / Accepted: date

Abstract Sensor networks, communication and financial networks, web and social networks are becoming increasingly important in our day-to-day life. They contain entities which may interact with one another. These interactions are often characterized by a form of autocorrelation, where the value of an attribute at a given entity depends on the values at the entities it is interacting with. In this situation, the collective inference paradigm offers a unique opportunity to improve the performance of predictive models on network data, as interacting instances are labeled simultaneously by dealing with autocorrelation. Several recent works have shown that collective inference is a powerful paradigm, but it is mainly developed with a fully-labeled training network. In contrast, while it may be cheap to acquire the network topology, it may be costly to acquire node labels for training. In this paper, we examine how to explicitly consider autocorrelation when performing regression inference within network data. In particular, we study the transduction of collective regression when a sparsely labeled network is a common situation. We present an algorithm, called CORENA (COllective REgression in Network dAta), to assign a *numeric* label to each instance in the network. In particular, we iteratively augment the representation of each instance with instances sharing correlated representations across the network. In this way, the proposed learning model is able to capture autocorrelations of labels over a group of related instances and feed-back the more reliable labels predicted by the transduction in the labeled network. Empirical studies demonstrate that the proposed approach can boost regression performances in several spatial and social tasks.

Keywords Collective Inference · Regression · Iterative Learning · Transduction

Corrado Loglisci · Annalisa Appice · Donato Malerba
Dipartimento di Informatica, Università degli Studi di Bari "Aldo Moro", via Orabona 4, I-70125 Bari, Italy
Tel.: +39-080-5443262 Fax: +39-080-5443269
E-mail: {corrado.loglisci, annalisa.appice, donato.malerba}@uniba.it

1 Introduction

With recent advances in pervasive computing trends, network data are becoming ubiquitous in our daily life. Examples include hypertext documents connected via hyperlinks, people connected via communication or social links, genes connected via co-regulation.

Regardless of where we encounter them, networks can be represented as graphs. They consist of entities (nodes), which may be connected with one another by links. The nodes in a network are generally of the same type and the links between nodes express various explicit relations. Information on the nodes is provided as a set of properties (attributes), whose values are associated with each node in the network. The links reflect the relation or dependence between the properties of the nodes. This is typically referred to as autocorrelation, that is, a cross-correlation of an attribute with itself (Cressie, 1993).

Autocorrelation is generally defined as deterministic or probabilistic dependence between the values of the same attribute on related instances (Epperson, 2000). This definition mirrors the way of thinking *pares cum paribus facillime congregantur* ("Like easily associates with Like" - Cicero). Autocorrelation is apparent in a wide variety of everyday situations, including spatial domains and social domains (Stojanova et al, 2012). In spatial domains, spatial autocorrelation is the cross-correlation of values of an attribute strictly due to their relatively close locations on a two-dimensional surface. Spatial autocorrelation exists when there is a systematic spatial variation in the values of a given property. This variation can exist in two forms, called positive and negative spatial autocorrelation (Legendre, 1993). In the positive case, the value of an attribute at a given location tends to be similar to the values of that attribute in nearby locations. This means that if the value of some attribute is low at a given location, the presence of spatial autocorrelation indicates that nearby values are also low. Conversely, negative spatial autocorrelation is characterized by dissimilar values at nearby locations. Goodchild (1986) remarks that positive autocorrelation is seen much more frequently in practice than negative autocorrelation in geophysical variables. This is justified by Tobler's first law of geography, according to which "everything is related to everything else, but near things are more related than distant things" (Tobler, 1970). In social domains, autocorrelation is recognized in the homophily principle, that is, the tendency of nodes with similar values to be linked with each other (McPherson et al, 2001). Homophily is observable, for example, in social networks where it is defined as the tendency of individuals to associate and bond with others who re similar (friendship). Actually, homophily shows that people's social networks are homogeneous with regard to many sociodemographic, behavioral, and intra-personal characteristics.

In this paper, we propose to perform regression inferences in network data by taking autocorrelation into account. We consider the scenario of sparsely labeled networks and describe a collective inference approach as an effective means to take autocorrelation into account. We illustrate an iterative con-

vergence algorithm that allows us to represent data instances as nodes of a network, learn the link structure of the nodes by modeling the autocorrelation of the descriptive information (node attributes) and predict node labels collectively with the goal of transduction. A regression model is iteratively learned from a partially labeled network. Accordingly to the philosophy of collective inference, the form of this model accounts for the autocorrelation of the labels of linked nodes. Accordingly to the philosophy of transductive inference, the model is learned, in order to reduce the inference error when predicting the labels for the remaining unlabeled network.

The paper is organized as follows. The next section clarifies the motivation and the actual contribution of this paper. Section 3 reports relevant related work. Section 4 describes the proposed algorithm. Section 5 describes the datasets, the experimental setup and reports the results. Finally, in Section 6 some conclusions are drawn and some future work is outlined.

2 Motivations and contributions

Collective inference is a fundamental approach to predictive inference in network domains (Getoor and Taskar, 2007). Traditional algorithms label data instances individually, regardless of the correlations or statistical dependencies that are prevalent in network data sets. In contrast, algorithms that reason collectively predict labels of linked instances simultaneously (Macskassy, 2007; Neville and Jensen, 2007; Macskassy and Provost, 2007; Gallagher et al, 2008; Sen et al, 2008). However, most work with collective inference performs learning using a fully-labeled training network. Unfortunately, in many situations gathering this information is tedious or expensive, and labeling large portions of the instances is infeasible (McDowell and Aha, 2012). In fact, this labeling may involve substantial human time and attention. On the other hand, learning with only a few such labels can lead to very poor performance (Shi et al, 2011b). In response, a few researchers (Bilgic et al, 2010; Shi et al, 2011b; McDowell and Aha, 2012) have recently investigated the collective inference paradigm in a partially labeled network, in order to produce the highest performance with the minimum number of labels.

The problem of learning with both labeled and unlabeled information is not novel. Two main settings have been proposed in the literature: the semi-supervised setting and the transductive setting (Seeger, 2001). The former is a type of inductive learning, since the learned function is used to make predictions on any possible example. The latter asks for less - it is only interested in reducing the inference error for the given set of unlabeled data, without improving the overall quality of the learned model. Since transduction needs no general hypothesis, it appears to be an easier problem than (semi-supervised) induction. McDowell and Aha (2012) have recently described the benefits of applying both these learning settings in two different network scenarios. Semi-supervised inference can be performed in across-network learning (Stojanova et al, 2012; McDowell and Aha, 2012), where a model is learned on one network

and then applied to a new disconnected network, with the goal of generalizing to other networks in the same domain. In contrast, transductive inference can be performed in within-network learning (Appice et al, 2009a; Steinhaeuser et al, 2011), where a model is learned on a partially labeled network and then applied to predict the class labels in the remainder of the network (i.e. the unlabeled portion).

The network scenario that we address in this work is the within-network, with some nodes labeled and some nodes unlabeled. Labels are numeric, sparse and scarce across the network. We consider the node attributes in a tight connection to the network structure. Thus connections (links in the network) between the data in the labeled/unlabeled set are used to deal with the autocorrelation property when generating the descriptive information included in the regression model. Therefore, in order to predict the value of the labels, besides the descriptive information we use the connections (links in the network) to the related/similar entities.

In this paper we have considered a number of challenges. First, in network domains, where instances exhibit the property of autocorrelation, the form of a regression model may depend on more than just the attributes of the instance itself. Thus, we consider the set of descriptive attribute values of the instance, as well as descriptive attribute values and labels of the linked instances. We define and compare several aggregation schema, in order to take linked instances into account. They are used to augment the nodes' attributes with new relational attributes. These new attributes model the autocorrelation of the linked values of the descriptive information, as well as of the labels. Second, it is difficult to learn accurate joint models from sparsely labeled networks. If learning methods ignore the unlabeled portion of the network, the model learned on one network can be applied to disconnected networks, with the goal of generalizing to other networks. However, in this situation, there may not be enough connectivity to learn the correlations accurately. Thus, we decide to incorporate the unlabeled portion of the network. This is done according to the principles of trasductive inference settings (Vapnik, 1998) - we predict labels, in order to reduce the inference error on the remaining unlabeled network, without improving the overall quality (e.g. generality) of the learned model. Third, it is difficult to estimate the reliability of predicted labels when collective inference is applied to numeric labels. We investigate the use of local measures of autocorrelation, in order to award labels which are predicted as part of a local pattern of autocorrelation. In particular, we consider local indicators of autocorrelation that return one value for each predicted label; this value expresses the degree to which that label is part of a cluster (i.e. the label is surrounded by similar labels). Fourth, the network structure reflecting the autocorrelation of the properties of the nodes is often hidden in the data instances. We investigate the computation of a dissimilarity measure, in order to determine this structure and inject it into the presented collective inference process. Nodes with similar descriptive values are recognized as part of a cluster and connected by links.

The algorithm presented in this paper is based on the preliminary work by Loglisci et al (2014). In our previous work, we proposed a collective regression algorithm that deals with the property of autocorrelation in spatial domains only - instances were assigned to spatial locations which were represented with spatial coordinates (e.g. latitude and longitude). We determined the network structure by using the spatial dimension of data only - instances closer in space were linked to each other, without accounting for correlations of descriptive information. We left out the case of sparsely labeled networks. The work presented here significantly extends the previous one in the following directions:

- Motivation for this work is given, both from the theoretical and application perspective.
- An extensive discussion of related work in Collective Inference is given.
- We generalize the algorithm to any network - the network structure is now determined by accounting for autocorrelation of descriptive information of instances.
- We consider the network regression task in a transductive formulation by exploring the case of sparsely labeled networks.
- We present new experiments on additional datasets, including real data about social networks that empirically confirm the considerations reported on the autocorrelation and show how our algorithm is able to reduce the prediction error by accounting for autocorrelation in network data.
- We now report experiments with sparsely labeled networks that illustrate how our algorithm adequately accounts for the effect of autocorrelation for regression goals in the case of the few labels that are known.

3 Related works

In this section, we review related studies on collective inference in network data. This learning paradigm has attracted significant attention in relational data mining (Jensen et al, 2004a). Network data is one typical type of relational data, while collective inference algorithms can exploit dependencies between instances. This makes collective inference one of the most favorable learning approaches for network data sets. Nevertheless, most work describes collective algorithms for network classification problems, while it overlooks regression problems.

Several collective classification approaches have been developed for a wide variety of real world applications, e.g. hyperlinked document classification and social network analysis. They can be roughly grouped into global algorithms and local algorithms (Sen et al, 2008). Global algorithms aim to train a classifier that seeks to optimize a global objective function often based on a Markov random field. They use Loopy belief propagation (Weiss, 2001; Taskar et al, 2002; Neville and Jensen, 2007; Sen et al, 2008) and Mean-field relaxation labeling (Weiss, 2001; Sen et al, 2008). These algorithms are usually computationally expensive, which limits their applicability to large-scale, real-world

network data. Local algorithms employ an iterative process whereby a local classifier predicts labels for each node, by using attributes of the nodes and relational attributes derived from the linked nodes. This type of approach involves an iterative process to update the labels and the relational attributes of the linked nodes, e.g. iterative convergence based approaches (Neville and Jensen, 2000; Getoor, 2005) and Gibbs sampling approaches (Jensen et al, 2004b). Both approaches are often combined with cautious inferences (McDowell et al, 2009).

The iterative convergence approaches are investigated in many studies (Neville and Jensen, 2000; Bilgic et al, 2007; McDowell et al, 2007; Fang et al, 2013). They account for the autocorrelation of labels and compute the label of a node depending on the labels of all its neighbors. In particular, iterative convergence approaches express a node by combining the node attributes and the relational attributes constructed by using the labels of all the neighbors of the node. The relational attributes can be computed by using an aggregation function over the neighbors, such as count, mode and proportion. Based on the node descriptive attributes and the relational attributes, an iterative convergence algorithm trains a classifier and iteratively updates the predictions of all nodes, by using the predictions for nodes with unknown labels. This process continues until the algorithm converges. Saha et al (2012) have recently described an iterative convergence algorithm to deal with multi-label classification problems. McDowell and Aha (2013) have shown that the accuracy of collective classification performed with both iterative convergence approaches and Gibbs sampling approaches may be increased by including, for each node, the descriptive attributes of the neighboring nodes as relational attributes. They conclude that using relational attributes built on both descriptive attributes and labels often produces the best accuracy. Finally, collective classification has been recently investigated in combination with active learning (Bilgic et al, 2010; Rattigan et al, 2007; Kuwadekar and Neville, 2011; Saha et al, 2014), as well as semi-supervised and transductive learning (Xiang and Neville, 2008; Shi et al, 2011a; McDowell and Aha, 2012).

Although collective classification has been widely investigated, collective regression has attracted a little attention. Chopra (2008) has defined a relational factor graph framework for performing regression in relational data. A single factor graph is used to capture dependencies among individual attributes of data instances, as well as dependencies among attributes associated with multiple data instances. The proposed models are learned with collective inferences by resorting to latent variables, in order to capture hidden inter-sample dependencies. However, these models are not formulated for the network settings. A few other approaches have been developed for the network regression task, but without resorting to the collective inference paradigm. Appice et al (2009b) described a transductive network regression algorithm developed in the co-training style. The algorithm is used within-network, in order to predict numeric labels of a sparsely labeled network. Two regression models are learned: the former using the descriptive attributes, the latter using the relational attributes constructed from the descriptive attributes. During an itera-

tive learning process, a regression model is used to label the unlabeled nodes for the other model. The autocorrelation of a label with labels of neighbor nodes is not explicitly accommodated when predicting the new labels. It is only used to estimate the reliability of predicted labels. In contrast, a supervised algorithm is illustrated by Stojanova et al (2012). It computes a final model that can be used across-network. The descriptive information (node attributes) and the network structure are used during the training phase, while only the descriptive information is used in the testing phase, where the network structure is disregarded, all testing examples are unlabeled and the network is not given. The autocorrelation of the labels is measured on nodes that are interconnected in the training network, but the model is not constrained by the structure of the training network.

4 Within-network collective regression

This section is devoted to the description of the algorithm CORENA that performs collective regression inferences in network data. CORENA inputs the nodes of a partially labeled network and predicts unknown targets by operating in two phases. First, it performs inferences on the autocorrelation of the descriptive data, in order to determine the link structure of the network. Then it uses an iterative convergence approach, in order to perform accurate collective inferences of the unknown targets. In the iterative phase, CORENA builds relational attributes that account for the property of autocorrelation of an attribute with itself over linked nodes. This produces an augmented vector of the descriptive attributes for each node. Similarly to McDowell and Aha (2013)'s work, the relational attributes are built from both the descriptive attributes and the target attribute associated with the nodes. Differently from McDowell and Aha (2013), relational attributes are built, in order to address a regression problem rather than a classification one. It is noteworthy that the relational descriptive attributes are computed once and for all before starting collective inferences. They are synthesized from the descriptive values which do not change during the iterative learning process. In contrast, the relational target attributes are updated at each new iteration, as new inferences on the linked targets can be made during the learning process. Finally, autocorrelation-aware reliability measures are considered, in order to identify the targets that are predicted with high reliability and feed them back into the network, in order to inform subsequent inferences about linked nodes. In the following, we first describe the network setting and the regression problem (Section 4.1), then we illustrate the algorithm that yields the collective regression inference within network data (Section 4.2). Finally, we analyze the time complexity of the presented algorithm (Section 4.3).

4.1 Network setting and regression problem

Assume we are given a dataset $D = (V, \mathbf{X}, Y, \omega)$, where V is a node set, each $\mathbf{x} \in \mathbf{X}$ is a vector of m descriptor (continuous or discrete) attributes for a node $v \in V$, each $y \in Y$ is a (possibly unknown) numeric target for v and ω is a dissimilarity threshold ($\omega \in \Re^+$). We are also given a set of *known* targets Y^L for nodes $V^L \subset V$ (labeled node set), such that $Y^L = \{y | v \in V^L\}$, while targets of $V^U = V - V^L$ (unlabeled node set) are unknown.

In this formulation, we assume the existence of a link structure that is implicit in the autocorrelation property of the data. A node $u \in V$ can be linked to a node $v \in V$ if the dissimilarity between the linked nodes is less than ω. In this way, given a mechanism to measure the dissimilarity between two nodes, we are able to derive a link structure $E \subseteq V \times V \times \Re^+$ from (V, ω), such that $E = \{\langle u, v, d\rangle | u, v \in V, d = diss(u, v)\ and\ d \leq \omega\}$. The pair (V, E) defines a network data setting for the regression problem formulation.

The regression problem is to receive full information (including labels) on the nodes of V^L, partial information (without labels) on the nodes of V^U ($V^U = V - V^L$), as well as the link structure E and predict the target values of V^U. It is noteworthy that this setting is the original distributional-free transductive setting proposed by Vapnik (1998),[1] which requires both the known set and the unknown set to be sampled from the node set V, without replacement. This means that, unlike the standard inductive setting, the nodes in the known (and unknown) set are supposed to be correlated, based on the existence of a link which (transitively) connects them.

4.2 The algorithm

The collective inference process is illustrated in Figure 1. CORENA inputs a node set V and the dissimilarity threshold ω. According to the problem formulation reported in Section 4.1, V comprises the labeled node set V^L and the unlabeled node set V^U (with $V = V^L \cup V^U$). The set V^L (see the red nodes in Figure 1) is spanned on both the descriptive space $\mathbf{X}$ and the target space Y, while V^U (see the blue nodes in Figure 1) is spanned on the descriptive space $\mathbf{X}$. CORENA computes a link structure E based on both V and ω (see phase 1 in Figure 1) and uses the network data (V, E), in order to output the targets $\hat{Y}^U$ predicted for the unlabeled node set V^U (see phase 2 in Figure 1).

4.2.1 Building the link structure

CORENA resorts to a dissimilarity-based approach, in order to build a link structure that depicts the property of autocorrelation through strongly correlated, linked nodes. This is done with the final aim of reducing the prediction

[1] Vapnik introduced an alternative transductive setting which is distributional, since both known set and unknown set are assumed to be drawn independently and identically from some unknown distribution. As shown in Vapnik (1998)(Theorem 8.1), error bounds for learning algorithms in the distribution-free setting apply to the more popular distributional transductive setting. This justifies our focus on the distributional-free setting.

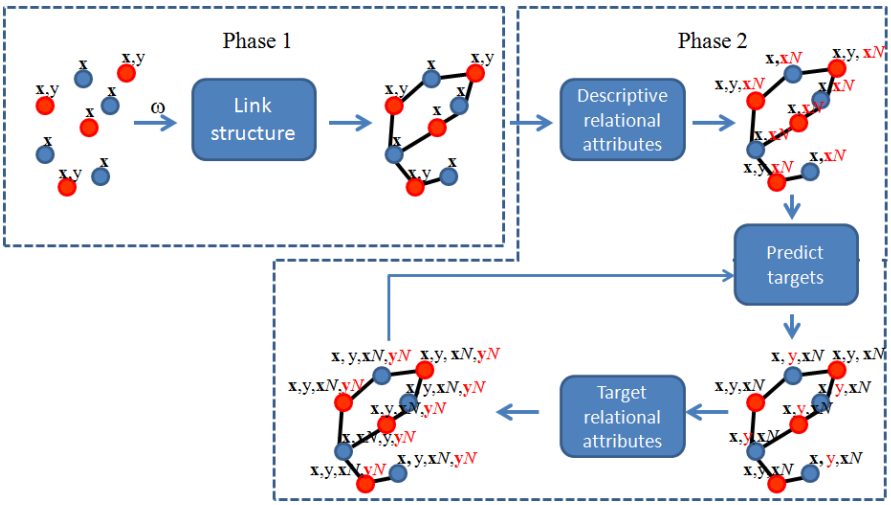

Fig. 1 The block diagram of the two-phase collective regression process performed by CORENA. In phase 1, CORENA determines the link structure of a partially labeled network. In phase 2, CORENA uses an iterative convergence approach, in order to collectively infer the unknown targets of the network.

error when collective inferences are performed on autocorrelation-aware linked nodes. It is based on the hypothesis that the autocorrelation of targets will be manifested jointly with the autocorrelation of descriptive values. Under this hypothesis, we look for the autocorrelation structure of the linked, frequently unknown, targets by measuring the similarity of the known descriptor values associated with.[2] The procedure is as follows: we build the link structure E from the node set V spanned on the descriptive space $\mathbf{X}$. We construct links that connect nodes measuring the nearest descriptive values over the node set. We assign a dissimilarity weight to each link, in order to measure the strength of the computed dissimilarity. The strength of the correlation between linked nodes can be estimated as the inverse of the power dissimilarity computed between the descriptive vectors associated with the nodes, so that the lower the dissimilarity weight, the higher the correlation strength of the link. In this paper, we use the Euclidean distance as a dissimilarity measure. The dissimilarity is computed after normalizing the values of each descriptive attribute $X \in \mathbf{X}$ in the interval [0,1]. For each candidate pair of nodes $(u, v) \in V \times V, u \neq v$, the dissimilarity weight $d = EuclideanDistance(u, v)$ is computed. The link (u, v, d) is added to E iff $d \leq \omega$.

4.2.2 Iterative convergence approach

[2] It is noteworthy that this phase can be overlooked when links are apriori defined in the input network data.

Algorithm 1 Iterative Convergence Approach$(V^L, V^U, E) \mapsto \hat{V}^U$

Require: V^L: the labeled node set spanned on $\mathbf{X} \times Y$
Require: V^U: the unlabeled node set spanned on $\mathbf{X}$
Require: E: the edge structure defined on $V \times V \times \Re^+$
Ensure: $\hat{V}^U$: the node set V^U labeled with the predicted targets $\hat{Y}^U$
1: iteration$\leftarrow 0$
2: $\mathbf{X}N \leftarrow$buildingRelationalAttributes$(V^L \cup V^U, E, \mathbf{X})$
3: $\hat{V}^U \leftarrow$ labeling$(V^U,$learnRegressionModel$(V^L, \mathbf{X} \times \mathbf{X}N \times Y))$ {Create a copy of V^U whose nodes are labeled with the targets predicted by the learned regression model}
4: $\mathbf{Y}N \leftarrow$buildingRelationalAttributes$(V^L \cup \hat{V}^U, E, Y)$
5: **repeat**
6: $\hat{V}^U_{new} \leftarrow$ labeling$(V^U,$learnRegressionModel$(V^L, \mathbf{X} \times \mathbf{X}N \times \mathbf{Y}N \times Y))$ {Create a copy of V^U whose nodes are labeled with the targets predicted by the learned regression model}
7: $noChange \leftarrow 0$
8: **for** $u \in V^U$ **do**
9: $oldR =$ computeReliability(getLabel$(u, \hat{V}^U), V^L)$
10: $newR =$ computeReliability(getLabel$(u, \hat{V}^U_{new}), V^L)$
11: **if** $newR > oldR$ **then**
12: labeling$(\hat{V}^U, u,$ getLabel$(u, \hat{V}^U_{new}))$
13: **else**
14: $noChange \leftarrow noChange + 1$
15: **end if**
16: **end for**
17: $iteration \leftarrow iteration + 1$
18: $\mathbf{Y}N \leftarrow$buildingRelationalAttributes$(V^L \cup \hat{V}^U_{new}, E, Y)$
19: **until** ($iteration = maxIt$ or $noChange \geq minNoChange$)

CORENA resorts to an iterative convergence approach, in order to collectively determine the unknown targets of V^U. The algorithm (see Algorithm 1) comprises an initialization phase and an iterative phase.

The initialization phase (Algorithm 1, lines 2-4) consists of three steps:

1. For each node in both the labeled and unlabeled set ($u \in V$), for each descriptive attribute ($X \in \mathbf{X}$), we build the associated relational attributes $\mathbf{X}N$ (Algorithm 1, line 2).
2. We learn a regression model from the labeled node set V^L spanned on $\mathbf{X} \times \mathbf{X}N \times Y$. This model is used to initialize the unknown targets of V^U. Predicted targets are now stored in $\hat{V}^U$ (Algorithm 1, line 3).
3. For each node in both the labeled and unlabeled set ($u \in V$), for the target attribute (Y), we build the associated relational attributes $\mathbf{Y}N$ (Algorithm 1, line 4).

In steps (1) and (3), we build the relational attributes by resorting to one of the attribute schemata described in Section 4.2.3.

The iterative phase is produced with the main loop (Algorithm 1, lines 5-19). We keep to the collective theory and look for new inferences that use "reliable" targets from previous inferences. The iterative phase consists of three steps:

1. We learn a new regression model from the labeled node set V^L, as it is spanned on the attribute space $\mathbf{X} \times \mathbf{X}N \times \mathbf{Y}N \times Y$. This model is used to

infer new targets for the unlabeled set V^U. These new targets are stored in $\hat{V}^U_{new}$ (Algorithm 1, line 6).
2. For each node of the unlabeled node set V^U, both the reliability of the target estimated at the previous iteration (and stored in $\hat{V}^U$) and the reliability of the target estimated in the current iteration (and stored in $\hat{V}^U_{new}$) are calculated. Reliability is quantified with a measure of local autocorrelation (see details in Section 4.2.4). For each node, the most reliable target is that maintained in $\hat{V}^U$ for the next iteration (Algorithm 1, lines 8-16).
3. The relational target attributes $\mathbf{Y}N$ are updated according to new reliable targets injected into $\hat{V}^U$ (Algorithm 1, line 4).

This iterative inference can stop in two cases. The maximum number of iterations $maxIt$ is reached or the number of predicted targets unchanged with respect to the previous iteration is greater than $minNoChange$. Both $maxIt$ and $minNoChange$ are user-defined parameters.

4.2.3 Computing relational attributes

Considering a base attribute A, we introduce three construction schemata according to which we can build a vector of relational attributes $\mathbf{A}N$ associated with A. The base attribute A can be either a descriptor attribute ($A \in \mathbf{X}$) or the target attribute ($A = Y$). In both cases, the constructed relational attributes are used to augment descriptive vectors associated with nodes of the network. All the schemata investigated in this study construct new descriptive attributes for the regression problem, by computing some summarization statistics of the base attribute A. New attributes contribute to handling the autocorrelation of the base attributes over local neighborhoods of the network data.

Definition 1 (Neighborhood) Let (V, E) be a network, u be a node ($u \in V$) and ν be the neighborhood radius ($\nu \in \Re^+$). The neighborhood $N(u)$ is a subset of V ($N(u) \subseteq V$) that includes all nodes $v \in V$, such that: $distance(u, v) \leq \nu$, where the $distance(\cdot, \cdot)$ is defined as follows:

$$distance(u, v) = \begin{cases} d & \text{if } (u, v, d) \in E \\ d + distance(z, v) & \text{if } \exists (u, z, d) \in E \\ & \textit{and } \exists\ path(z, v, E) \\ \infty & \text{otherwise} \end{cases}. \quad (1)$$

According to Formula 1, we distinguish three cases: (1) u and v are directly linked, meaning there is a link between them (2) u and v are transitively linked, meaning there is a path between them and (3) u and v are disconnected, meaning there is no path between them. In the first case, we output the dissimilarity weight associated with the direct link. In the second case, we output the sum of dissimilarity weights calculated over the least-cost path. This is computed with Dijkstra's algorithm (Dijkstra, 1959), which is a

graph search algorithm that solves the single-source shortest path problem for a graph with non-negative edge path costs, producing a shortest path tree. In the third case, the dissimilarity is infinity.

Attribute Schema ***Var1*** Given the base attribute A, we build two new relational attributes, $AN(mean)$ and $AN(stDev)$, based on A. Both attributes are computed by aggregating A over the neighborhoods constructed with radius $\nu = \omega$ throughout the network. Let u be a node, $AN(u, mean)$ is computed as the weighted mean of the values of A falling in the neighborhood $N(u)$, that is:

$$AN(u, mean) = \frac{\sum_{v \in N(u)} \lambda(u, v) \times val(A)}{\sum_{v \in N(u)} \lambda(u, v)}, \tag{2}$$

where $\lambda(u, v) = \frac{1}{distance(u,v)}$ with $distance(\cdot, \cdot)$ computed as reported in Formula 1, and $val(A)$ is the value of the attribute A for the node v. $aN(u, stDev)$ is the standard deviation of the values of A falling in the neighborhood $N(u)$.

Attribute Schema ***Var2*** By following the idea investigated by Ohashi and Torgo (2012), as well as Appice et al (2013), we use two neighborhoods, $N1(u)$ with radius $\nu = \Omega$ and $N2(u)$ with radius $\nu = 2\Omega$, respectively. Given the base attribute A, we compute five new relational attributes, $AN(mean, 1)$ and $AN(mean, 2)$, $AN(stDev, 1)$, $AN(stDev, 2)$ and $AN(speed)$. For each node u, $AN(u, mean, 1)$ and $AN(u, mean, 2)$, $AN(u, stDev, 1)$ and $AN(u, stDev, 2)$ are calculated as described in **Var1** by using $N1(u)$ (for $AN(u, mean, 1)$ and $AN(u, stDev, 1)$) and $N2(u)$ (for $AN(u, mean, 2)$ and $AN(u, stDev, 2)$) respectively as neighborhood units of analysis. $AN(speed)$ is calculated, in order to represent the speed at which values of A change when stepping back from u over the network. Formally, this is computed as follows:

$$AN(u, speed) = \frac{AN(u, mean, 1)}{AN(u, mean, 2)}. \tag{3}$$

Attribute Schema ***Var3*** Given the base attribute A that assumes d distinct values, we build d new relational attributes. These attributes represent the frequency histogram of A, as it is computed on the neighborhoods constructed with radius $\nu = \omega$ throughout the network. In practice, we build one relational attribute for each distinct value of A. Let u be a node, val be a distinct value of A, $AN(u, val)$ is computed as the frequency of val over the neighborhood $N(u)$. We note that if A is a numeric attribute, we discretize this attribute before computing the relational attributes associated with it. The discretization is done by resorting to the equal frequency discretization algorithm.

4.2.4 Measuring reliability of targets

We measure the reliability of the targets predicted at each iteration, in order to select reliable targets that are fed back to the training network for the next iteration. Intuitively, reliable targets should manifest the property of autocorrelation, so that similar targets can be plausibly propagated to linked nodes. The higher the autocorrelation of a target with linked targets, the more reliable the target in the network. Two local measures of autocorrelation, namely the Anselin Local Moran's Index LMI (Arthur, 2008) and the standardized Getis and Ord local GI^* (Anselin, 1995), are used. Both are originally defined in spatial statistics. They are used in the analysis on sensor networks (Appice and Malerba, 2014), but can be easily adapted to a general-purpose network data setting. They return one numeric value per node. This value denotes the degree of autocorrelation of a value over linked nodes. For each unlabeled node, we compute the measure of local autocorrelation for both the target predicted in the present iteration and the target output at the previous iteration. The more reliable targets are conserved for the next iteration. In this study, we compute the autocorrelation of a predicted target with respect to the real targets of the originally labeled node set (V^L).

The Anselin Local Moran's Index LMI is used in spatial statistics to distinguish between local patterns of positive autocorrelation and negative autocorrelation. In this study, we compute this measure, in order to quantify the reliability of the target $\hat{Y}$ predicted at the unlabeled node $u \in V^U$. Formally,

$$LMI(u) = \left(\frac{(u(\hat{Y}) - \overline{Y})}{m2}\right) \sum_{v \in V^L} \big(\lambda(u,v) \times (val(Y) - \overline{Y})\big), \tag{4}$$

where $u(\hat{Y})$ is the target predicted at the node u, $\overline{Y}$ and $m2$ are respectively the mean and the second moment of the targets in the labeled set V^L, $\lambda(u,v) = \frac{1}{distance(u,v)}$ with $distance(\cdot,\cdot)$ computed as reported in Formula 1. A positive value for $LMI(u)$ indicates that $u(Y)$ is linked to similar values (positive autocorrelation). A negative value for $LMI(u)$ indicates that $u(Y)$ is linked to dissimilar values (negative autocorrelation). The higher the $LMI(u)$, the more reliable the predicted target $u(\hat{Y})$.

The standardized Getis and Ord local GI^* is used in spatial statistics to identify objects that are part of clusters of high/low values. Contrary to LMI, this measure assumes positive autocorrelation, while it does not capture the presence of negative autocorrelation. It is computed as follows:

$$GI*(u) = \frac{\left(\sum_{v \in V^L, u \neq v} \lambda(u,v)u(\hat{Y}) - \overline{Y}\ \Lambda(u)\right)}{\sqrt{\frac{m2}{n-1}\left(n \sum_{v \in V^L, v \neq u} \lambda(u,v)^2 - \overline{\Lambda(u)}^2\right)}}, \tag{5}$$

where $\overline{\Lambda(u)} = \sum_{v \in V^L, u \neq v} \lambda(u,v)$ and n in the cardinality of V^L. The interpretation of $GI*$ is different from that of LMI: a significant positive value indicates that the predicted target is a high value linked to high values, while a significant negative value indicates that the predicted value is a low value linked to low values. Therefore, the higher the absolute value of $GI^*(u)$, the more reliable the predicted target $u(\hat{Y})$.

4.3 Time complexity analysis

Let n^L be the number of labeled nodes (i.e. $n^L = |V^L|$), n^U be the number of unlabeled nodes (i.e. $n^U = |V^U|$), such that $n = n^L + n^U$, e be the number of links (i.e. $e = |E|$), m be the number of descriptive attributes (i.e. $m = |\mathbf{X}|$) and l be the average number of links coming from a node. The computational complexity of CORENA depends on the cost of building the link structure E from the node set V, as well as the cost of performing collective inferences over the network data (V, E). The time cost of building the link structure E is $\mathrm{O}(n^2 m)$ as the Euclidean distance is computed on the descriptive vectors of each pair of nodes of V. The time cost of performing the iterative convergence learning process in fact depends on the complexity of: (1) determining the dissimilarity weights of the least-cost paths between each pair of (transitively) linked nodes (see Formula 1),[3] (2) building the vectors of the relational descriptive attributes $\mathbf{X}N$, (3) learning a regression model, (4) predicting the unknown targets $\hat{Y}^U$ and measuring the reliability of predicted targets and (5) constructing the vector of the relational target attributes YN. Additionally, steps 3-5 are iterated per $maxIt$ number of times, in the worst case. More specifically, the determination of the low-cost paths from one node to all nodes of a network has a time cost $\mathrm{O}(e+n\log n)$ when the implementation of Dijkstra's algorithm is used based on a min-priority queue implemented by a Fibonacci heap. On the other hand, the construction of the vector of relational descriptive attributes associated with a base attribute has a time cost $\mathrm{O}(l)$, with the attribute schema **Var1**, $\mathrm{O}(l^2)$, with the attribute schema **Var2** and $\mathrm{O}(l\log l + l)$, with the attribute schema **Var3**, assuming an optimal algorithm is used for sorting when the equal frequency discretization is performed. The discovery of a regression model has a time cost $\mathrm{O}(\mathcal{L})$, where $\mathcal{L}$ depends on the base learning algorithm used in this phase. Finally, the evaluation the reliability of the predicted targets has a time cost $\mathrm{O}(n^U \times n^L)$. Therefore, the time cost of the entire process is

$$\mathrm{O}\left(\underbrace{n^2}_{E} + \underbrace{n(e+n\log n)}_{low\ cost\ path} + \underbrace{ml^2}_{\mathbf{X}N} + maxIt \times \left(\underbrace{\mathcal{L}}_{base\ regression} + \underbrace{n^U \times n^L}_{reliability} + \underbrace{l^2}_{YN}\right)\right).$$

[3] The dissimilarity weights associated with the least-cost paths can be pre-computed before starting the iterative learning. As they depend on the descriptive values, they do not change over the collective inferences.

Table 1 Data description. For each dataset we report the domain of the attributes, the number of nodes - n, the number of descriptive attributes - m

	Domain	N	M
LAB	Spatial	52	5
NWE	Spatial	970	6
MS	Spatial	420	6
MF	Spatial	420	6
NOAA-clwvi	Spatial	270	34
NOAA-pr	Spatial	270	34
Movies	Social	415	13
VideoL	Social	752	9

5 Experimental evaluation and discussion

CORENA is written in Java and evaluated in several real-world data sets. Experiments are run on an Intel DualCore CPU @2.00GHz desktop running Windows 7 Professional. Before we proceed to present empirical results, we describe both the network data used and the experimental settings.

5.1 Network data

In this experimental evaluation, we use eight real data collections, acquired from spatial and social domains. The characteristics of the datasets are summarized in Table 1. In each data collection, a number of descriptive attributes, as well as the numeric target, which will be discussed in detail below, are associated with the nodes. In spatial domains, nodes are spatial. They are represented by geographic coordinates (latitude and longitude), so that they are at some geographic distance apart in space. In this situation, we extend the set of descriptive attributes of each node with the geographic coordinates of the node. This allows us to account for the spatial dimension of data when computing the dissimilarity measure $diss()$ and determining the links of the network structure.

The Intel Berkeley Lab dataset (**LAB**) (Intel Berkeley Lab, 2004) collects in-door measurements of humidity, light and voltage values (descriptive attributes) and temperature (target) transmitted every 31 seconds from 54 sensors deployed in the Intel Berkeley Research lab, between February 28th and April 5th 2004. The network is built by considering the sensors as nodes. The attributes associated with the nodes are computed as the mean of values measured between February 28th and March 21st, 2004.

The North-West Census dataset (**NWE**) contains census data provided by the 1998 Census and collected in the European project SPIN! (May and Savinov, 2003). The dataset includes measures of deprivation level in the ward (the census unit), including index scores such as the Jarman Underprivileged Area score, Townsend score, Carstairs score and the Department of the Environment score (descriptive attributes), as well as the percentage of mortality (target). The network is built by considering the 970 wards as nodes.

The datasets SIGMEA MS and SIGMEA MF (**MS** and **MF**) (Demšar et al (2005)) are derived from one multi-target dataset containing measurements of pollen dispersal (crossover) rates from two lines of plants (target): the transgenic male-fertile (MF) and the non-transgenic male-sterile (MS) lines of oilseed rape. The descriptive attributes are the cardinal direction and distance of the sample point from the center of the donor field, the visual angle between the sample plot and the donor field, the shortest distance between the plot and the nearest edge of the donor field. Both networks are built by considering the 817 sampling points as nodes.

The NOAA data are collected for climatology studies (Simons, 2011). Here we consider two datasets, namely **NOAA-clwvi** and **NOAA-pr**. They are derived from the content of the condensed water in the clouds (target) and the precipitation flux (target) respectively. In both datasets, we use 31 descriptive attributes concerning meteorology, heat flux, pressure, temperature and wind. Data are measured daily from 270 stations distributed worldwide. The network is built by considering the stations as nodes. The attributes associated with the nodes are computed as the mean of values measured between October and November, 2000.

The **Movies** dataset contains movie ratings given to movies by users of the online movie recommender service Movielens, collected during the period 1997-1998 (Grouplens, 1998). Specifically, for each movie, the dataset contains the IMDB movie identifier, genre, country, movie director and filming location, as well as all/top/audience critics ratings: average scores, numbers of reviews/fresh scores/rotten scores from the Rotten Tomatoes film review aggregator. The target is all the critics' ratings: all other rating data are not included in the analysis. Similarly to Stojanova et al (2012), we are interested in pairs of movies that are ranked together by a single user, where the selected users rated at least 20 movies. The network structure has 500 nodes corresponding to the movies (labeled with their properties).

The **VideoL** dataset contains the ECML PKDD 2011 Discovery Challenge data (Antulov-Fantulin et al, 2011). The data are related to the content of VideoLectures.net, a free and open access multimedia repository of video lectures, mainly on research and educational topics. The target is the total number of views of lectures published online, where pairs of lectures are viewed together (not necessarily consecutively) with at least two distinct cookie-identified browsers. The descriptive attributes include several properties of a lecture, such as the type, category, author and language of the lecture, as well as the recording and publishing dates of the lecture. Similarly to Stojanova et al (2012), we use the complete data from the Challenge for 2009. The network structure has 754 nodes containing the lectures along with their properties.

5.2 Experimental setup

5.2.1 Evaluation metrics

We evaluate the accuracy performance of several variants of CORENA and compare it to that of some competitor algorithms. The evaluation is performed on the collection of datasets described above. The accuracy is measured in terms of the RMSE and estimated by using the inverse 10-fold cross validation (Malerba et al, 2009). For each trial, the compared algorithms are trained on a single (labeled) data instance and tested on the hold-out nine data instances, forming the unlabeled set. This experimental design, that uses small training set sizes, allows us to validate the transductive approach.

5.2.2 Compared algorithms

In all experiments, we use M5' (Wang and Witten, 1997) as a base learner.[4] It is noteworthy that this choice does not exclude the possibility of using any other regression method as a base learner of our transductive approach. We run CORENA by setting $minNoChange$ equal to half the size of the unlabeled node set and $maxIt$ equal to 15.

We first evaluate the accuracy of CORENA using different variable schemata, reliability measures and dissimilarity thresholds. This comparative study aims to understand the impact of these parameters on the accuracy of the transductive process. In our experiments, the variable schema ranges between **Var1**, **Var2** and **Var3** (see details in Section 4.2.3), while the reliability measure ranges between **LMI** and **GI*** (see details in Section 4.2.4). We consider 10 bins in the equal-frequency discretization of **Var3**. We select the dissimilarity threshold ω (see details in Section 4.1) depending on the connectivity degree we intend to test in the network. This selection is done by exploring a set of candidate thresholds with a grid search procedure. The search is tailored, in order to determine approximately the lowest value of ω that allows us to build a connected network with each node linked to at least $MinLinks$ nodes. We evaluate CORENA by setting $MinLinks$ as a percentage ($MinLinks\% = 30\%, 45\%$) of the entire node set size.

Then we compare the accuracy performance of CORENA to that of the base learner M5', to the network regression algorithm NCLUS (Stojanova et al, 2012), as well as to its traditional ancestor CLUS (Blockeel et al, 1998). For this comparison, we consider the several variants of CORENA tested in this study, while the default parameter configuration is considered for the competitors.

5.2.3 Statistical comparison

In order to compare the predictive capabilities of the learned models, we use the non-parametric Wilcoxon two-sample paired signed rank test (Orkin and

[4] We use the Java implementation of M5' included in the WEKA toolkit (Witten and Frank, 2005). We consider the default configuration setup with the pruning option enabled.

Drogin (1990)). To perform the test, we assume that the experimental results of the two algorithms compared are independent pairs $\{(q_1, r_1), (q_2, r_2), \ldots, (q_n, r_n)\}$ of sample data. We then rank the absolute value of the differences $q_i - r_i$. The Wilcoxon test statistics WT^+ and WT^- are the sum of the ranks from the positive and negative differences, respectively. We test the null hypothesis H_0: "no difference in distributions" against the two-sided alternative H_1: "there is a difference in distributions". Intuitively, when $WT^+ \gg WT^-$ and vice versa, H_0 is rejected. Whether WT^+ should be considered "much greater than" WT^- depends on the significance level considered. The null hypothesis of the statistical test is that the two populations have the same continuous distribution. Since, in our experiments, q_i and r_i are Avg.MSEs, $WT^+ \gg WT^-$ implies that the second method (R) is better than the first (Q). In all experiments reported in this empirical study, the significance level used in the test is set at 0.05.

5.3 Results and discussion

Table 2 reports the average RMSE of the several variants of CORENA, the base learner M5', as well as the competitors NCLUS and CLUS. The accuracy of CORENA is evaluated by varying the variable schema, the reliability measure and the connectivity threshold percentage. The analysis of these results lead to several considerations.

First, we analyze the accuracy performance of CORENA along the variable schemata **Var1**, **Var2** and **Var3**. We observe that **Var3** shows, in general, better accuracy performance (e.g. the lowest error) than **Var1** and **Var2**. In particular, **Var3** outperforms both **Var1** and **Var2** in 6 out of 8 trials with **LMI** and $MinLinks\% = 30\%$, 5 out 8 trials with **GI*** and $MinLinks\% = 30\%$, 5 out of 8 trials with **LMI** and $MinLinks\% = 45\%$ and 7 out of 8 trials with **GI*** and $MinLinks\% = 45\%$. The reason for this result can be found in the information which is represented by the relational attributes constructed. Both **Var1** and **Var2** build relational attributes by calculating basic statistics such as mean and standard deviation. Therefore, constructed attributes summarize linked data without paying great attention to how the target values are differently distributed among the single nodes. In contrast, **Var3** builds relational attributes that describe the histogram of the target values over the linked data. In this way, constructed attributes are able to quantify possible changes in the distribution of target values. Final considerations concern the fact that social networks (Movies and VideoL) often perform a "higher" drop of accuracy than spatial networks when either **Var1** or **Var2** are used in place of **Var3** (see 3154 (**Var2**) vs 1.36(**Var3**) with Movies, **LMI** and $MinLinks\% = 30\%$, 1599 (**Var2**) vs 1.37(**Var3**) with Movies, **GI*** and $MinLinks\% = 30\%$, 1971 (**Var1**) vs 751(**Var3**) with VideoL, **LMI** and $MinLinks\% = 30\%$, 2058 (**Var1**) vs 758(**Var3**) with VideoL, **GI*** and $MinLinks\% = 30\%$, 2.54 (**Var2**) vs 1.324(**Var3**) with Movies, **LMI** and $MinLinks\% = 45\%$, 2.75 (**Var2**) vs 1.321(**Var3**) with Movies, **GI*** and $MinLinks\% = 45\%$, 2873 (**Var2**)

vs 690(**Var3**) with VideoL, **LMI** and $MinLinks\% = 45\%$, 1279 (**Var2**) vs 696(**Var3**) with VideoL, **GI*** and $MinLinks\% = 45\%$). This may depend on the fact that the variable constructors of both **Var1** and **Var2** have been inspired by the data aggregation indicators efficaciously used by Ohashi and Torgo (2012), as well as Appice et al (2013), but in spatial and spatio-temporal domains only. They probably work in an improper manner in social domains.

Second, we analyze the influence of the reliability measures **LMI** and **GI*** on the accuracy performance of CORENA. We observe that **LMI** generally outperforms **GI***. In particular, **LMI** outperforms **GI*** in 6 out of 8 trials with **VAR1** and $MinLinks\% = 30\%$, 6 out of 8 trials with **VAR2** and $MinLinks\% = 30\%$, 8 out of 8 trials with **VAR3** and $MinLinks\% = 30\%$, 7 out of 8 trials with **VAR1** and $MinLinks\% = 45\%$, 7 out of 8 trials with **VAR2** and $MinLinks\% = 45\%$ and 5 out of 8 trials with **VAR3** and $MinLinks\% = 45\%$. Both reliability measures are based on the computation of local indicators of autocorrelation, that is, Anselin's Local Moran Index (**LMI**) and Getis and Ord Local Index (**GI***), respectively. We ascribe the best performance of **LMI** in this study to the ability of Anselin's Local Moran Index to detect both positive and negative autocorrelations. Getis and Ord Local Index does not capture the presence of negative autocorrelations, while it can distinguish clusters of high and low values. However, this study shows that knowing if a label is part of a cluster of high/low value is subordinate to knowing if the label is part of a cluster (positively autocorrelated with linked labels) or if it is an outlier (negatively autocorrelated with a linked node).

Third, we analyze the influence of the connectivity threshold percentage $MinLinks\%$. We observe that the accuracy performance of CORENA is often improved when augmenting the number of links over the network (i.e. when increasing $MinLinks\%$). In particular, CORENA with $MinLinks\% = 45\%$ outperforms CORENA with $MinLinks\% = 30\%$ in 5 out of 8 trials with **VAR1** and **LMI**, 6 out of 8 trials with **VAR2** and **LMI**, 4 out of 8 trials with **VAR3** and **LMI**, 4 out of 8 trials with **VAR1** and **GI***, 6 out of 8 trials with **VAR2** and **GI*** and 6 out of 8 trials with **VAR3** and **GI***. It is noteworthy that augmenting the connectivity threshold percentage, more and more nodes are linked to each other over the network. In any case, similarity-aware weights are associated with the nodes, thus we are able to quantify the real strength of the correlation between the linked nodes. Our study shows that learning by accounting for weights associated with links allows CORENA to be robust in the presence of possible links relating uncorrelated nodes. To complete the analysis of how the network connectivity influences the learning process, we analyze the ratio of the computation time of CORENA when $MinLinks\% = 45\%$ to the computation time of CORENA when $MinLinks\% = 30\%$. The results, collected here for the several variants of CORENA, are plotted in Figures 2(a)-2(h). They show that the higher the connectivity over the network, the slower, in general, the learning process. We can also observe that the magnitude of this trend changes with the network, but it is stable if analyzed either along the variable schema or along the reliability measure. In any case, this analysis highlights that the trade-off between accuracy and efficiency is an

open issue in networked collective inference. The emerging research problem, that requires further investigation in the future, is that of determining the minimum connectivity over the network, in order to gain the highest accuracy by spending the lowest computation time.

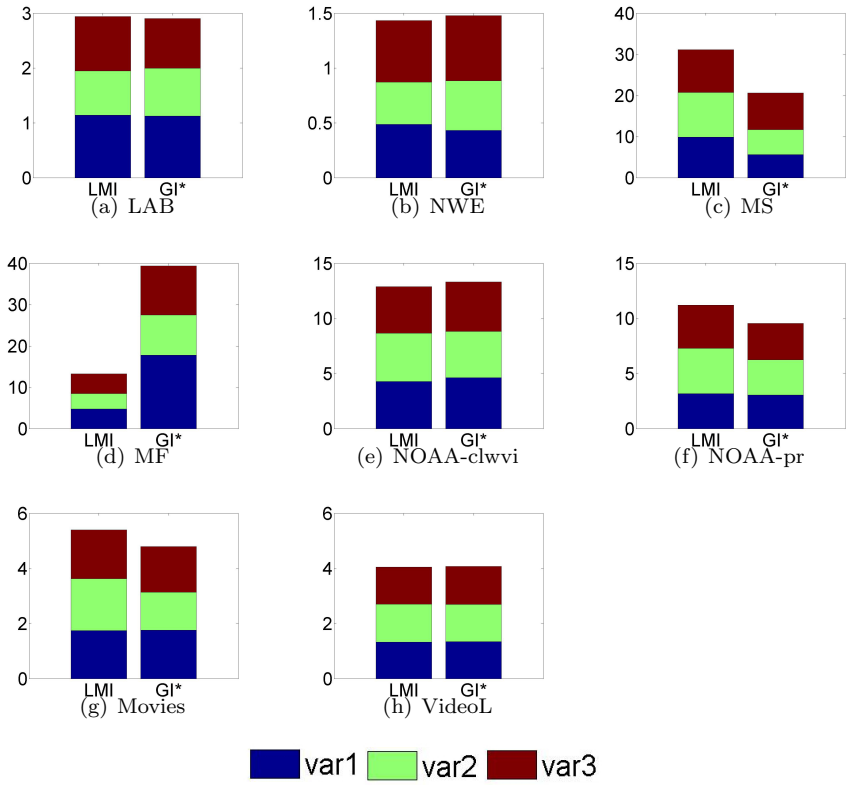

Fig. 2 Analysis of the computation time of CORENA along $MinLinks\%$: the ratio of the computation time of CORENA when $MinLinks\% = 45\%$ to the computation time of CORENA when $MinLinks\% = 30\%$.

Finally, we analyze the accuracy performance of CORENA compared to that of both the baseline learner M5' and the competitors NCLUS and CLUS. Results show that there is always one variant of CORENA that performs better than the baseline learner (without collective inference), as well as the competitors. The only exception is the dataset NWE when the network is computed with $MinLinks\% = 45\%$. In this case, all variants of CORENA perform slightly worse than the base learner, although sligtly better than the competitors. This analysis is confirmed by the results of the pairwise Wilcoxon signed rank test reported in Tables 3 - 5, which compare the accuracy performance of CORENA to that of M5', NCLUS and CLUS. In particular, statistical test results in Table 3 confirm that collective inference gains accuracy by improv-

ing, in general, the base learner. On the other hand, results in Tables 4 and 5 show that CORENA is usually (statistically) better than its competitors NCLUS and CLUS, especially when the variable schema **VAR3** is used, in order to construct the relational variables. This result is particularly interesting for NCLUS, as this is the competitor that explicitly addresses a network regression problem.

6 Conclusion

Many regression problems involve network data that exhibit the property of autocorrelation. Techniques for collective inference allow us to naturally handle the property of autocorrelation by increasing the learning accuracy in network data problems. However, they usually require a fully labeled network and apply to classification problems. On the contary, we address the problem of regression and consider the case of sparsely labeled networks. This is an important task as demonstrated by the use of real world datasets.

The network setting that we address is the transductive one. We use both the descriptive and the target information of the labeled node set, the descriptive information of the unlabeled node set, as well the link structure of the network, in order to determine collectively the numeric targets of the unlabeled part of the network. We describe an iterative convergence algorithm that accounts for the property of autocorrelation in both the descriptive space and the target space, in order to derive the link structure of the network, synthesize new descriptive relational attributes and estimate the reliability of the predicted targets. The regression model is iteratively learned from the labeled node set that is spanned on the descriptive space, augmented with the relational attributes. At each iteration, the most reliable targets are injected into the network and used to update the relational attributes associated with them, as well as the regression model.

We evaluate the accuracy of our approach in an extensive set of real world problems of network regression in the areas of spatial and social networks. Empirical evaluation investigates the influence of the variable schema, the reliability measure and the link structure on the performances of the presented algorithm. In addition, it compares the performance of our algorithm to that of traditional regression algorithms (M5', CLUS), which disregard the network structure, as well as a network regression algorithm (NCLUS), which accounts for network structure and autocorrelation as well. Results show that our algorithm outperforms competitors, although our approach gains higher accuracy when the relational attributes are computed by resorting to the frequency aggregator (variable schema **Var3**) and the reliable targets are identified by resorting to the Anselin Local Moran Index of autocorrelation (reliability measure LMI). Finally, we observe that the empirical evaluation reveals that augmenting the number of linked nodes over the network generally produces higher accuracy of predicted targets, although this at the expense of the time cost.

Several directions for further work still are to be explored. The trade-off between accuracy and efficiency is an open issue that requires further investigation, in order to apply this algorithm to big data problems. The link structure is actually determined according to a global, user-defined threshold. The automated, local determination of this parameter deserves immediate attention. The local estimation should allow us to handle sparsely dense networks. In a similar fashion, one might consider selecting an appropriate autocorrelation measure for the reliability estimation, as well as an appropriate variable schema for the relational attribute construction. Finally, it would be interesting to investigate solutions of active learning when selecting the nodes to be labeled over the network.

Acknowledgments

This work fulfills the research objectives of the PON 02_00563_3470993 project "VINCENTE - A Virtual collective INtelligenCe ENvironment to develop sustainable Technology Entrepreneurship ecosystems" funded by the Italian Ministry of University and Research (MIUR), as well as the ATENEO 2012 project "Mining Complex Patterns" funded by University of Bari Aldo Moro. The authors wish to thank Antonella Montinari for her support in developing the software, Saso Dzeroski for providing SIGMEA data and Lynn Rudd for her help in reading the manuscript.

References

Anselin L (1995) Local indicators of spatial association:lisa. Geographical Analysis 27(2):93–115

Antulov-Fantulin N, Bošnjak M, Žnidaric M, Grčar M, Morzy M, Šmuc T (2011) Discovery challenge overview. In: ECML-PKDD 2011 Discovery Challenge Workshop, Springer, pp 7–20

Appice A, Malerba D (2014) Leveraging the power of local spatial autocorrelation in geophysical interpolative clustering. Data Mining and Knowledge Discovery 28(5-6):1266–1313

Appice A, Ceci M, Malerba D (2009a) An iterative learning algorithm for within-network regression in the transductive setting. In: Gama J, Costa VS, Jorge AM, Brazdil P (eds) Discovery Science, 12th International Conference, DS 2009, Springer, Lecture Notes in Computer Science, vol 5808, pp 36–50

Appice A, Ceci M, Malerba D (2009b) An iterative learning algorithm for within-network regression in the transductive setting. In: Discovery Science, Springer, pp 36–50

Appice A, Pravilovic S, Malerba D, Lanza A (2013) Enhancing regression models with spatio-temporal indicator additions. In: Proceedings of the 13rd International Conference of the Italian Association for Artificial Intelligence on Advances in Artificial Intelligence, AI*IA 2013, Springer, Lecture Notes in Computer Science, vol 8249, pp 433–444

Arthur G (2008) A history of the concept of spatial autocorrelation: A geographer's perspective. Geographical Analysis 40(3):297–309

Bilgic M, Namata GM, Getoor L (2007) Combining collective classification and link prediction. In: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops, ICDMW 2007, IEEE Computer Society, pp 381–386

Bilgic M, Mihalkova L, Getoor L (2010) Active learning for networked data. In: Fürnkranz J, Joachims T (eds) Proceedings of the 27th International Conference on Machine Learning, ICML 2010, Omnipress, pp 79–86

Blockeel H, Raedt LD, Ramon J (1998) Top-down induction of clustering trees. In: Shavlik JW (ed) Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998, Morgan Kaufmann, pp 55–63

Chopra SP (2008) Factor graphs for relational regression. ProQuest

Cressie N (1993) Statistics for Spatial Data, 1st edn. Wiley

Demšar D, Debeljak M, Lavigne C, Džeroski S (2005) Modelling pollen dispersal of genetically modified oilseed rape within the field. In: Abstracts of the 90th ESA Annual Meeting, The Ecological Society of America, p 152

Dijkstra EW (1959) A note on two problems in connexion with graphs. Numerische mathematik 1(1):269–271

Epperson B (2000) Spatial and space-time correlations in ecological models. Ecological modeling 132:63–76

Fang M, Yin J, Zhu X (2013) Transfer learning across networks for collective classification. In: Proceedings of the 13th International Conference on on Data Mining, ICDM 2013, IEEE Computer Society, pp 161–170

Gallagher B, Tong H, Eliassi-Rad T, Faloutsos C (2008) Using ghost edges for classification in sparsely labeled networks. In: Proc. 14th ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining, ACM, pp 256–264

Getoor L (2005) Link-based classification. In: Advanced Methods for Knowledge Discovery from Complex Data, Advanced Information and Knowledge Processing, Springer London, pp 189–207

Getoor L, Taskar B (2007) Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press

Goodchild M (1986) Spatial autocorrelation. Geo Books

Grouplens (1998) `http://www.grouplens.org/node/12`

Intel Berkeley Lab (2004) `http://db.csail.mit.edu/labdata/labdata.html`

Jensen D, Neville J, Gallagher B (2004a) Why collective inference improves relational classification. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, New York, NY, USA, KDD '04, pp 593–598, DOI 10.1145/1014052.1014125, URL `http://doi.acm.org/10.1145/1014052.1014125`

Jensen D, Neville J, Gallagher B (2004b) Why collective inference improves relational classification. In: Proc. 10th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, ACM, pp 593–598

Kuwadekar A, Neville J (2011) Relational active learning for joint collective classification models. In: Getoor L, Scheffer T (eds) Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Omnipress, pp 385–392

Legendre P (1993) Spatial autocorrelation: Trouble or new paradigm? Ecology 74(6):1659–1673

Loglisci C, Appice A, Malerba D (2014) Collective inference for handling autocorrelation in network regression. In: Andreasen T, Christiansen H, Talavera JCC, Ras ZW (eds) Foundations of Intelligent Systems - 21st International Symposium, ISMIS 2014, Springer, Lecture Notes in Computer Science, vol 8502, pp 542–547

Macskassy S, Provost F (2007) Classification in networked data: a toolkit and a univariate case study. Machine Learning 8:935–983

Macskassy SA (2007) Improving learning in networked data by combining explicit and mined links. In: Proc. 22nd Intl. Conf. on Artificial Intelligence, AAAI Press, pp 590–595

Malerba D, Ceci M, Appice A (2009) A relational approach to probabilistic classification in a transductive setting. Eng Appl of AI 22(1):109–116, DOI 10.1016/j.engappai.2008.04.005, URL `http://dx.doi.org/10.1016/j.engappai.2008.04.005`

May M, Savinov AA (2003) Spin!-an enterprise architecture for spatial data mining. In: Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, KES 2003, Part I, pp 510–517

McDowell L, Aha DW (2012) Semi-supervised collective classification via hybrid label regularization. In: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Omnipress

McDowell L, Aha DW (2013) Labels or attributes?: rethinking the neighbors for collective classification in sparsely-labeled networks. In: He Q, Iyengar A, Nejdl W, Pei J, Rastogi R (eds) Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, ACM, pp 847–852

McDowell L, Gupta KM, Aha DW (2007) Case-based collective classification. In: Wilson D, Sutcliffe G (eds) Proceedings of the 20th International Florida Artificial Intelligence Research Society Conference, AAAI Press, pp 399–404

McDowell L, Gupta KM, Aha DW (2009) Cautious collective classification. Journal of Machine Learning Research 10:2777–2836

McPherson M, Smith-Lovin L, Cook J (2001) Birds of a feather: Homophily in social networks. Annual Review of Sociology 27:415–444

Neville J, Jensen D (2000) Iterative classification in relational data. In: Proc. 17th Intl. Joint Conf. on Artificial Intelligence, AAAI Press

Neville J, Jensen D (2007) Relational dependency networks. Journal of Machine Learning Research 8:653–692

Ohashi O, Torgo L (2012) Wind speed forecasting using spatio-temporal indicators. In: ECAI 2012, IOS Press, vol 242, pp 975–980

Orkin M, Drogin R (1990) Vital Statistics. McGraw Hill

Rattigan M, Maier M, Jensen D (2007) Exploiting network structure for active inference in collective classification. In: Seventh IEEE International Conference on Data Mining - ICDM Workshops 2007., pp 429–434

Saha T, Rangwala H, Domeniconi C (2012) Multi-label collective classification using adaptive neighborhoods. In: Proceedings of the 11th International Conference on Machine Learning and Applications, ICMLA 2012, vol 1, pp 427–432

Saha T, Rangwala H, Domeniconi C (2014) FLIP: active learning for relational network classification. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2014, Part III, Springer, Lecture Notes in Computer Science, vol 8726, pp 1–18

Seeger M (2001) Learning with labeled and unlabeled data. Tech. rep.

Sen P, Namata G, Bilgic M, Getoor L, Gallagher B, Eliassi-Rad T (2008) Collective classification in network data. AI Magazine 29:3:93–106

Shi X, Li Y, Yu P (2011a) Collective prediction with latent graphs. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM 2011, ACM, pp 1127–1136

Shi X, Li Y, Yu PS (2011b) Collective prediction with latent graphs. In: Macdonald C, Ounis I, Ruthven I (eds) Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, ACM, pp 1127–1136

Simons RA (2011) Erddap - the environmental research division's data access program. http://coastwatchpfegnoaagov/erddap Pacific Grove, CA: NOAA/NMFS/SWFSC/ERD

Steinhaeuser K, Chawla NV, Ganguly AR (2011) Complex networks as a unified framework for descriptive analysis and predictive modeling in climate science. Statistical Analysis and Data Mining 4(5):497–511

Stojanova D, Ceci M, Appice A, Dzeroski S (2012) Network regression with predictive clustering trees. Data Min Knowl Discov 25(2):378–413

Taskar B, Abbeel P, Koller D (2002) Discriminative probabilistic models for relational data. In: Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence, UAI 2002, Morgan Kaufmann Publishers Inc., pp 485–492

Tobler W (1970) A computer movie simulating urban growth in the Detroit region. Economic Geography 46(2):234–240

Vapnik V (1998) Statistical Learning Theory. Wiley

Wang Y, Witten I (1997) Induction of model trees for predicting continuous classes. In: Proc. Poster Papers of the European Conference on Machine Learning, Faculty of Informatics and Statistics, University of Economics, Prague, pp 128–137

Weiss Y (2001) Comparing the mean field method and belief propagation for approximate inference in mrfs. In: Opper M, Saad D (eds) Advanced Mean Field Methods, MIT Press, pp 229–243

Witten I, Frank E (2005) Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco

Xiang R, Neville J (2008) Pseudolikelihood em for within-network relational learning. In: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM 2008, IEEE, pp 1103–1108

Table 2 Average RMSE (estimated by inverse 10-fold CV): CORENA vs M5, NCLUS and CLUS. Results for NWE are multiplied by 10^3. The lowest error is in bold. (*) denotes the variable schema with the lowest error for CORENA, when both the reliability measure and the percentage $minLinks\%$ are selected. ω denotes the similarity threshold as it is computed by performing the grid search on the set of candidate thresholds with the selected $MinLinks\%$.

			CORENA						M5'	NCLUS	CLUS
				LMI			**GI***				
	$MinLinks\%$	ω	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**			
LAB	30%	0.07	2.89	2.89	**2.39***	3.01	2.92	2.4*	8.98	12.4	12.4
NWE		0.07	2.47	2.49	**2.46***	2.49	2.51	**2.46***	2.47	2.5	2.61
MS		0.07	24.56	10.53	**6.81***	29.32	11.34	6.95*	6.95	8.27	7.76
MF		0.08	5.47	2.76	**2.75***	5.74	2.83	2.81*	2.85	3.23	3.01
NOAA-clwvi		0.15	33146	**31544***	32820	32912	32759*	33472	34037	66245	46145
NOAA-pr		0.15	6.01	10.4	**3.68***	5.94	5.93*	11.27	**3.68**	29.82	20.6
Movies		0.15	**1.3***	3154	1.36	1.31*	1599	1.37	2.62	12.81	12.81
VideoL		0.15	1971	761	**751***	2058	798	758*	2257	760	760
LAB	45%	0.08	2.60	2.62	**2.38***	2.62	2.81	2.4*	8.98	12.4	12.4
NWE		0.08	2.48	2.49*	2.49*	2.5	2.5	2.49*	**2.47**	2.5	2.61
MS		0.15	6.88	6.87	**6.77***	6.96	6.91	6.86*	6.86	8.27	7.76
MF		0.2	2.78	**2.76***	2.89	2.78	2.77*	2.89	2.85	3.23	3.01
NOAA-clwvi		0.2	33334*	33898	33901	33293	33912	**31482***	34037	66245	46145
NOAA-pr		0.2	3.688	3.69	3.684*	3.68	3.69	**3.67***	3.688	29.8	20.6
Movies		0.22	1.323*	2.524	1.324	1.333	2.75	**1.321***	2.26	12.81	12.81
VideoL		0.2	745.7	2873	**690.6***	752	1279	696*	2257	760.3	760.3

Table 3 Pairwise Wilcoxon signed rank test comparing accuracy performance (RRMSE) of CORENA to that of M5'. + means that CORENA is better than M5' (i.e. WT+ > WT-), - means that M5' is better than CORENA (i.e. WT+ < WT-). (++) and (–) report results in the case H0 (hypothesis of equal performance) is rejected at the 0.05 significance level.

CORENA	$MinLinks\% = 30\%$						$MinLinks\% = 45\%$					
vs	**LM**			**GI***			**LM**			**GI***		
M5'	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**
LAB	++	++	++	++	++	++	++	++	++	++	++	++
NWE	-	-	+	-	-	+	-	-	-	-	-	-
MS	-	-	+	-	-	-	-	-	+	-	-	-
MF	-	+	+	-	+	+	+	-	-	+	+	-
NOAA-clwvi	+	+	+	+	+	+	+	+	+	+	+	+
NOAA-pr	-	-	+	-	-	-	+	+	+	-	-	+
Movies	++	-	++	++	-	++	++	++	++	++	++	++
VideoL	+	+	+	+	+	++	+	+	++	+	-	++

Table 4 Pairwise Wilcoxon signed rank test comparing accuracy performance (RRMSE) of CORENA to that of NCLUS. + means that CORENA is better than NCLUS (i.e. WT+ > WT-), - means that NCLUS is better than CORENA (i.e. WT+ < WT-). (++) and (–) report results in the case H0 (hypothesis of equal performance) is rejected at the 0.05 significance level.

CORENA	$MinLinks\% = 30\%$						$MinLinks\% = 45\%$					
vs	**LM**			**GI***			**LM**			**GI***		
NCLUS	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**
LAB	++	++	++	++	++	++	++	++	++	++	++	++
NWE	+	+	++	+	+	++	++	+	+	+	+	+
MS	-	–	++	-	–	++	++	++	++	++	++	++
MF	-	+	++	-	+	++	++	++	++	++	++	++
NOAA-clwvi	++	++	++	++	++	++	++	++	++	++	++	++
NOAA-pr	++	++	++	++	++	++	++	++	++	++	++	++
Movies	++	–	++	++	–	++	++	++	++	++	++	++
VideoL	-	-	+	-	-	+	+	-	++	+	-	++

Table 5 Pairwise Wilcoxon signed rank test comparing accuracy performance (RMSE) of CORENA to that of CLUS. + means that CORENA is better than CLUS (i.e. WT+ > WT-), - means that CLUS is better than CORENA (i.e. WT+ < WT-). (++) and (–) report results in the case H0 (hypothesis of equal performance) is rejected at the 0.05 significance level.

CORENA	$MinLinks\% = 30\%$						$MinLinks\% = 45\%$					
vs	**LM**			**GI***			**LM**			**GI***		
CLUS	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**	**Var1**	**Var2**	**Var3**
LAB	++	++	++	++	++	++	++	++	++	++	++	++
NWE	++	++	++	++	++	++	++	++	++	++	++	++
MS	-	-	+	-	-	+	+	+	+	+	+	+
MF	-	+	+	-	+	+	+	+	+	+	+	+
NOAA-clwvi	++	++	++	++	++	++	++	++	++	++	++	++
NOAA-pr	++	++	++	++	++	++	++	++	++	++	++	++
Movies	++	-	++	++	-	++	++	++	++	++	++	++
VideoL	-	-	+	-	-	+	+	-	++	+	-	++