Collective Matrix Factorization of Predictors, Neighborhood and Targets for Semi-supervised Classification

Lucas Rego Drumond[1], Lars Schmidt-Thieme[1], Christoph Freudenthaler[1], and Artus Krohn-Grimberghe[2]

[1] ISMLL - Information Systems and Machine Learning Lab
University of Hildesheim, Germany
http://www.ismll.uni-hildesheim.de
[2] AIS-BI University of Paderborn, Germany
{ldrumond,schmidt-thieme,freudenthaler}@ismll.de, artus@aisbi.de

Abstract. Due to the small size of available labeled data for semi-supervised learning, approaches to this problem make strong assumptions about the data, performing well only when such assumptions hold true. However, a lot of effort may have to be spent in understanding the data so that the most suitable model can be applied. This process can be as critical as gathering labeled data. One way to overcome this hindrance is to control the contribution of different assumptions to the model, rendering it capable of performing reasonably in a wide range of applications. In this paper we propose a collective matrix factorization model that simultaneously decomposes the predictor, neighborhood and target matrices (PNT-CMF) to achieve semi-supervised classification. By controlling how strongly the model relies on different assumptions, PNT-CMF is able to perform well on a wider variety of datasets. Experiments on synthetic and real world datasets show that, while state-of-the-art models (TSVM and LapSVM) excel on datasets that match their characteristics and have a performance drop on the others, our approach outperforms them being consistently competitive in different situations.

Keywords: Semi-supervised classification; factorization models.

1 Introduction

In certain domains, the acquisition of labeled data might be a costly proccess making it difficult to exploit supervised learning models. In order to surmount this, the field of semi-supervised learning [2] studies how to learn from both labeled and unlabeled data. Given the small amount of available labeled data, semi-supervised learning methods need to make strong assumptions about the data distribution. The most prominent assumptions (briefly discussed in Section 2) are the cluster and the manifold assumption.

It is usually the case that, if such assumptions do not hold, unlabeled data may be seriously detrimental to the algorithms performance [3]. As a consequence, determining in advance good assumptions about the data and developing or choosing

V.S. Tseng et al. (Eds.): PAKDD 2014, Part I, LNAI 8443, pp. 286–297, 2014.

models accordingly may be as critical as gathering labeled data. In chapter 21 from [2] an extensive benchmark evaluation of semi-supervised learning approaches on a variety of datasets is presented resulting in no overall winner, i.e. no method is consistently competitive at all datasets, so that one has to rely on background knowledge about the data. General models that can work well on different kinds of data offer a means to circumvent this pitfall. One promising family of models that work in this direction are factorization models [10]. Such models are flexible enough to fit different kinds of data without overfitting (given that they are properly regularized). However, to the best of our knowledge, there is no systematic evaluation of the capabilities of factorization models as semi-supervised classifiers.

In this work, we show how semi-supervised learning can be approached as a factorization problem. By factorizing the predictor matrix, one can exploit unlabeled data to learn meaningful latent features together with the decision boundary. This approach however is suboptimal if the data is not linearly separable. Thus, we enforce neighboring points in the original space to still be neighbors in the learned latent space by factorizing also the adjacency matrix of the nearest neighbor graph. We call this model the Predictor/Neighborhood/Target Collective Matrix Factorization (PNT-CMF). While the state-of-the-art approaches may usually be very effective in some datasets, they perform poorly in others; we provide empirical evidence that PNT-CMF can profit from unlabeled data, making them competitive in settings where different model assumptions hold true. The main contributions of the paper are:

- We propose PNT-CMF, a novel model for semi-supervised learning that collectively factorizes the predictor, neighborhood and target relation.
- We devise a learning algorithm for PNT-CMF that is based on simultaneous stochastic gradient descent over all three relations.
- In experiments on both synthetic and real-world data sets we show that our approach PNT-CMF outperforms existing state-of-the-art methods for semi-supervised learning. Especially we show that while existing approaches work well for datasets with matching characteristics (cluster-like datasets for TSVM and manifold-like datasets for LapSVM), our approach PNT-CMF consistently performs competitive under varying characteristics.

2 Related Work

For a thorough survey of literature on semi-supervised learning in general, the reader is referred to [2] or [15]. In order to learn from just a few labeled data points, the models have to make strong assumptions about the data. One can categorize semi-supervised classification methods according to such assumptions. Historically, the first semi-supervised algorithms were based on the idea that, if two points belong to the same cluster, they have the same label. This is called the *cluster* assumption. If this assumption holds, it is reasonable to expect that the optimal decision boundary should stay in a *low density region*. Methods which fall into this category are the transductive SVMs [5] and the information regularization framework [11]. As pointed out by [15], this assumption does not hold

true if, for instance, the data is generated by two highly overlapping gaussians. In this case, a *generative model* like the EM with mixture models [8] would be able to devise an appropriate classifier.

The second most relevant assumption is that data points lie on a low dimensional manifold [1]. One can think of the *manifold* assumption as the cluster assumption on the manifold. A successful approach implementing this assumption is the class of algorithms based on manifold regularization [1] [7], which regularizes the model by forcing points with short geodesic distances to have similar values for the decision function. Since the geodesic distances are computed based on the laplacian of a graph representation of the data, these methods can also be regarded as graph-based methods. This class of semi-supervised algorithms define a graph where the nodes are the data points and the edge weights are the similarity between them and are regularized to force neighboring nodes to have similar labels. Graph based methods like the one based on Gaussian Fields and Harmonic functions [16] and global consistency method [14] rely on the laplacian of the similarity graph to achieve this. These methods can be seen as special cases of the manifold regularization framework [1].

All of those methods have shown to be effective when their underlying assumptions hold true. However, when this is not the case, their performance might actually be worsened by unlabeled data. The factorization models proposed here are more flexible regarding the structure of the data since (i) they do not assme decision function lies in a low density region, but map the features to a space where they are easily separable instead and (ii) enforces neighboring points to have the same label by co-factorizing the nearest neighbor matrix, which contribution to the model can be adjusted so that the model is robust to datasets where this information is not relevant. Multi-matrix factorization as predictive models have been investigated by [10]. Previous work on the semi-supervised learning of factorization models has either focused on different tasks or had different goals from this work. While we are here focused on semi-supervised classification, previous work has focused on other tasks like clustering [12] and non-linear unsupervised dimensionality reduction [13]. A closer match is the work from Liu et al. [6] which approaches multi-label classification. Their method relies on ad-hoc similarity measures for instances and class labels that should be chosen for each kind of data. While their method only works for multi-label cases, the approach presented here deals with binary classification.

3 Problem Formulation

In a traditional supervised learning problem, data are represented by a predictors matrix X, where each row represents an instance predictor vector $\mathbf{x_i} \in \mathbb{R}^{|\mathcal{F}|}$, $\mathcal{F}$ being the set of predictors, and a target matrix Y, with each row $\mathbf{y_i}$ containing the values of the target variables for the instance i. Depending on the task, Y can take various forms. Since throughout the paper we will consider the binary classification setting, we assume Y to be a one dimensional matrix (i.e. a vector) $\mathbf{y} \in \{-1, +1\}^{|\mathcal{I}|}$, where $\mathcal{I}$ the set of instances.

Generally speaking, a learning model uses some training data $D^{\text{Train}} := (X^{\text{Train}}, \mathbf{y}^{\text{Train}})$ to learn a model that is able to predict the values in some test data $\mathbf{y}^{\text{Test}}$ given X^{Test}, all unseen when learning.

In the semi-supervised learning scenario, there are two distinct sets of training instances: the first comes with their respective labels, i.e. X_L^{Train} and $\mathbf{y}_L^{\text{Train}}$; the second is composed of training instances for which their respective labels are not known during training time, i.e. X_U^{Train}, thus the training data is composed by $D^{\text{Train}} := (X_L^{\text{Train}}, X_U^{\text{Train}}, \mathbf{y}_L^{\text{Train}})$.

At this point learning problems can again be separated in two different settings. In some situations, it is known during the learning time which instances should have their labels predicted. This is called *transductive* learning [4]. In other situations however, the focus is on learning a general model able to make predictions based on instances unknown during learning, which is called the *inductive* setting.

4 Factorization Models for Semi-supervised Classification

4.1 Classification as a Multi-matrix Factorization Task

Factorization models [10] decompose and represent a matrix as a product of two factor matrices $X \approx f(V, H)$ where $f : \mathbb{R}^{n \times k} \times \mathbb{R}^{m \times k} \to \mathbb{R}^{n \times m}$ is a function representing how two factors can be used to reconstruct the original matrix $X \in \mathbb{R}^{n \times m}$. One common choice for f is $f_X(V, H) := VH^\top$.

In some cases, two or more matrices need to be factorized at the same time. [10] propose a loss function for decomposing an arbitrary number of matrices. Be $\mathcal{M}$ the set of matrices to be factorized and Θ the set of factor matrices to be learned, each matrix $M \in \mathcal{M}$ is reconstructed by $M \approx f_M(\theta_{M_1}, \theta_{M_2})$, where $\theta_{M_1}, \theta_{M_2} \in \Theta$. Thus the overall loss is

$$J(\Theta) := \sum_{M \in \mathcal{M}} \alpha_M \, l_M(M, f_M(\theta_{M_1}, \theta_{M_2})) + Reg(\Theta) \tag{1}$$

where l_M is a loss function that measures the reconstruction error of matrix M, $0 \leq \alpha_M \leq 1$ is a hyperparameter defining the relative importance of each loss for the general objective function, and Reg is some regularization function.

In a classification problem the predictor matrix X and their respective targets $\mathbf{y}$ are given. A factorization model approximates X as a function of two latent factor matrices V, H, i.e. $X \approx f_X(V, H)$ and $\mathbf{y}$ as a function of $\mathbf{w} \in \mathbb{R}^{1 \times k}$ and V, i.e. $\mathbf{y} \approx f_y(V, \mathbf{w})$. This way each row $\mathbf{v}_i$ of matrix V is a k-dimensional representation of the instance $\mathbf{x}_i$. The predicted targets are found in the approximating reconstruction of $\mathbf{y}$, i.e. $\mathbf{y} \approx V\mathbf{w}^\top$. This task can be defined as finding the factor matrices V, $\mathbf{w}$ and H that optimize a specific case of equation 1, namely:

$$J(V, \mathbf{w}, H) := \alpha\, l_X(X, f_X(V, H)) + (1 - \alpha) l_{\mathbf{y}}(\mathbf{y}, f_{\mathbf{y}}(V, \mathbf{w})) + Reg(V, \mathbf{w}, H) \quad (2)$$

For the purposes of this work, the approximation functions will be the product of the factor matrices, i.e.:

$$X \approx f_X(V, H) = VH^\top$$
$$\mathbf{y} \approx f_y(V, \mathbf{w}) = V\mathbf{w}^\top$$

This model allows for different possible choices for l_X and l_y. Since it is common to represent instances of a classification problem as real valued feature vectors (and this is the case for the datasets used in our experiments), we used the squared loss as l_X. For $l_{\mathbf{y}}$, a number of losses are suited for classification problems. We use the hinge loss here, since we are dealing with binary classification problems. In principle, any loss function can be used, so the one that best fits the task at hand should be selected.

One drawback of using the hinge loss is that it is not smooth, meaning that it is less easy to optimize. To circumvent this, we use the smooth hinge loss proposed by [9]:

$$h(y, \hat{y}) := \begin{cases} \frac{1}{2} - y\hat{y} & \text{if } y\hat{y} \leq 0, \\ \frac{1}{2}(1 - y\hat{y})^2 & \text{if } 0 < y\hat{y} < 1, \\ 0 & \text{if } y\hat{y} \geq 1 \end{cases} \quad (3)$$

4.2 Neighborhood Based Feature Extraction

The model presented so far is flexible enough for fitting a variety of datasets, but it still can not handle non-linear decision boundaries unless a very high number of latent dimensions is used, which are difficult to estimate from few labeled data. On the top of that, if the data follow the manifold assumption (i.e. data points lying next to each other on the manifold tend to have the same labels), the factorization model presented so far, will not be able to exploit this fact to learn better decision boundaries. Because we use a linear reconstruction of $\mathbf{y}$, if the data is not linearly separable in the learned latent space, the algorithm will fail to find a good decision boundary. This problem can be circumvented by forcing that the nearest-neighborhood relationship is maintained on the latent space. This works because the factorization of $\mathbf{y}$ forces the labeled points from different classes to be further apart from each other in the latent space. Forcing that the neighborhood in the original space is preserved in the latent space will make the unlabeled points to be "dragged" towards their nearest labeled neighbor, thus separating clusters or structures in the data, making it easier to find a good decision boundary that is linear in the latent space.

To accomplish this we construct a nearest neighbor graph and factorize its adjacency matrix $K \in \mathbb{R}^{n \times n}$, where each position k_{ij} is 1 if instance j is one of the N nearest neighbors of i and 0 otherwise. The nearest neighbor matrix

is reconstructed as $K \approx f_K(V, V)$. This enforces that instances close to each other have similar latent features, thus being close in the latent space. We call this model the *Predictor/Neighborhood/Target Collective Matrix Factorization (PNT-CMF)*. Complexity control is achieved using Tikhonov regularization. The objective function we optimize PNT-CMF for in this work is shown in eq. 4, where $||\cdot||_F$ stands for the Frobenius norm.

$$J(V, H, \mathbf{w}) := \alpha_X \sum_{i \in \mathcal{I}} \sum_{f \in \mathcal{F}} (x_{i,f} - \mathbf{v}_i \mathbf{h}_f^\top)^2 + \alpha_K \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} (k_{i,j} - \mathbf{v}_i \mathbf{v}_j^\top)^2$$
$$+ \alpha_Y \sum_{i \in \mathcal{L}} h(y_i, \mathbf{v}_i \mathbf{w}^\top) + \lambda_V ||V||_F^2 + \lambda_W ||\mathbf{w}||_F^2 + \lambda_H ||H||_F^2 \quad (4)$$

The hyperparameter α_K controls the importance of the nearest neighbor relation to the model. This is a very important parameter since that, if the data does not have a cluster structure, or if it has a misleading cluster structure (i.e. points belonging to different classes in the same cluster), the factorization of K will harm the model more than help to improve its performance.

We point out that factorizing the nearest neighbor matrix is related to, but differs from, the concept of manifold regularization [1]. Manifold regularization forces instances close to each other to have similar values in the decision function. Forcing that neighbors have similar latent features causes the same effect in our multi-matrix factorization model. It has been observed however that, if the data does not follow the manifold or cluster assumption, manifold regularization based methods like Laplacian SVM and Laplacian Regularized Least Squares fail to find a good solution [2]. In this case the best solution is to set the laplacian regularization constant to zero, reducing the model to a fully supervised one, not taking advantage of the unlabeled data points. Here, by setting $\alpha_K = 0$ one still has a powerful semi-supervised model that simply does not rely on the neighborhood relation. In the experiments conducted in this paper we have some indication that it is possible to automatically estimate good values for α_K through model selection without any background knowledge on the dataset, although further investigation in this direction is needed.

4.3 Semi-supervised Learning of PNT-CMF

In a transductive setting, the test instances predictors are available at training time. A factorization model can naturally make use of this information by adding those predictors to the X matrix. If $\mathbf{y}$ is only partially observed, then the training data for the transductive factorization model is

$$X := \begin{bmatrix} X^{\text{Train}} \\ X^{\text{Test}} \end{bmatrix}, \mathbf{y} := \begin{bmatrix} \mathbf{y}^{\text{Train}} \\ ? \end{bmatrix}$$

Here non-observed values are denoted by a question mark. Learning a transductive model means optimizing a factorization model for equation 4 on the data above.

Algorithm 1. LearnPNT-CMF

procedure LEARNPNT-CMF
input: $X \in \mathbb{R}^{n \times m}, \mathbf{y} \in \mathbb{R}^{n \times l}$, $\lambda_V, \lambda_H, \lambda_{\mathbf{w}}, \alpha_X, \alpha_Y, \alpha_K, d, N$

$V, H, \mathbf{w} \sim \mathcal{N}(0, \sigma)$
$\mathcal{I} := \{1, ...n\}$
$\mathcal{F} := \{1, ...m\}$
$\mathcal{L} := \{1, ..., l\}$
K $\leftarrow$ computeNearestNeighborMatrix(X, d, N)
repeat
  draw i from $\mathcal{I}$ and f from $\mathcal{F}$
  $\mathbf{v_i} \leftarrow \mathbf{v_i} + \mu \left(\alpha_X \frac{\partial}{\partial \mathbf{v}_i} l_X(x_{i,f}, \mathbf{v}_i \mathbf{h}_f^\top) + \lambda_V \mathbf{v_i} \right)$
  $\mathbf{h_f} \leftarrow \mathbf{h_f} + \mu \left(\alpha_X \frac{\partial}{\partial \mathbf{h}_f} l_X(x_{i,f}, \mathbf{v}_i \mathbf{h}_f^\top) + \lambda_H \mathbf{h_f} \right)$
  draw i, j from $\mathcal{I}$, such that $i \neq j$
  $\mathbf{v_i} \leftarrow \mathbf{v_i} + \mu \left(\alpha_K \frac{\partial}{\partial \mathbf{v_i}} l_K(k_{ij}, \mathbf{v}_i \mathbf{v}_j^\top) + \lambda_V \mathbf{v_i} \right)$
  $\mathbf{v_j} \leftarrow \mathbf{v_j} + \mu \left(\alpha_K \frac{\partial}{\partial \mathbf{v_j}} l_K(k_{ij}, \mathbf{v}_i \mathbf{v}_j^\top) + \lambda_V \mathbf{v_j} \right)$
  draw i from $\mathcal{L}$
  $\mathbf{v_i} \leftarrow \mathbf{v_i} + \mu \left(\alpha_y \frac{\partial}{\partial \mathbf{v_i}} l_{\mathbf{y}}(y_i, \mathbf{v}_i \mathbf{w}^\top) + \lambda_V \mathbf{v_i} \right)$
  $\mathbf{w} \leftarrow \mathbf{w} + \mu \left(\alpha_y \frac{\partial}{\partial \mathbf{w}} l_{\mathbf{y}}(y_i, \mathbf{v}_i \mathbf{w}^\top) + \lambda_W \mathbf{w} \right)$
until convergence
return $V, H, \mathbf{w}$
end procedure

To learn this model, a stochastic gradient descent algorithm is applied as shown in Algorithm 1. The algorithm starts by randomly initializing the parameters to be learned. The values for each one of them are drawn from a normal distribution with mean 0 and variance 0.001. Following this, the neighborhood is computed based on a distance measure d. In this paper we used the euclidean distance $d(\mathbf{x}_i, \mathbf{x}_j) := ||\mathbf{x}_i - \mathbf{x}_j||^2$. Next the parameters are updated in the direction of the negative gradient of each loss.

4.4 Learning Inductive Factorization Models for Classification

The same model could be used in an inductive setting, where the instances for which we want to make predictions are not known during training time. This can be achieved by simply setting X and $\mathcal{I}$ arguments in Algorithm 1 to X^{Train} and $\mathcal{L}$ respectively. In this process, the training data for the inductive model is

$$X := X^{\text{Train}}, \mathbf{y} := \mathbf{y}^{\text{Train}}$$

One drawback of this approach is that, in order to make out-of-sample predictions, the latent representations of the test instances need to be inferred for each test instance separately. In other words, for a previously unseen instance $\mathbf{x}_i$, its respective latent feature vector $\mathbf{v}_i$ is not computed during the training phase. We approach this problem by adding a fold-in step after learning the model.

The fold-in step takes a new instance x_i and maps it to the same latent feature space as the training instances. One straight-forward way to accomplish this is to minimize Equation 5.

$$\arg\min_{\mathbf{v}_i} J(\mathbf{x}_i, H, \mathbf{v}_i) := \alpha_X \, ||\mathbf{x}_i - \mathbf{v}_i H^\top||^2 + \alpha_K \, ||\mathbf{k}_i - \mathbf{v}_i V^\top||^2 + \lambda_V ||\mathbf{v}_i||^2 \quad (5)$$

5 Evaluation

The main goals of the experiments are: 1.) compare PNT-CMF against state-of-the-art semi-supervised classifiers; 2.) assess the robustness and competitiveness of our factorization model across datasets with different characteristics; 3.) observe how useful semi-supervised transductive factorization models are compared to their inductive supervised counterparts (i.e. we want to observe how much can factorization models benefit from unlabeled data).

5.1 Datasets

Chapelle et al. [2] have run an extensive benchmark analysis of semi-supervised learning models on 6 datasets, which are also used here. The *g241c*, *g241d*, *digit* and *usps* datasets have all 1500 instances and 241 predictors while the *bci* dataset, 400 instances and 117 predictors. Finally the *text* dataset has 1500 instances and 11960 predictors. The task for all data sets is binary classification. Here we note that we conduct experiments only on binary classification in order not to contaminate the results with influences from different tasks. The application and evaluation of the model on other tasks like multi-class and multi-label classification and regression is left for future work.

5.2 Setup

For the evaluation of the methods proposed here, we employed the same protocol used in the benchmark analysis presented by [2]. Each dataset, comes with two different sets of splits: the first one with 10 randomly chosen labeled training instances and the second with 100, each one with 12 splits. We used exactly the same splits as [2], which are available for download[1]. Each model was evaluated on a transductive semi-supervised setting. In order to be able to answer to question 3 posed in the beginning of this section, we also evaluated the models on a fully supervised setting.

The performance of the model was measured using the hinge loss since it is the measure the evaluated models are optimized for. We also evaluated the models on AUC but the results are suppressed here due to the lack of space. The findings from the experiments on both measures where however very similar.

[1] `http://olivier.chapelle.cc/ssl-book/benchmarks.html`

5.3 Baselines

We compare PNT-CMF against representative methods implementing the two most important assumptions of semi-supervised learning. As a representative of the manifold assumption we chose the Laplacian SVM (LapSVM) trained in the primal [7], since manifold regularization has been one of the most successful approaches for semi-supervised learning.

The second baseline is the transductive version of SVMs (TSVM) [5] which implements the low density (or cluster) assumption. On the inductive case, TSVM reduces to a standard SVM. Besides being representatives of their working assumptions, both LapSVM and TSVM optimize the same loss used for PNT-CMF in this paper. Other graph based methods [16][14] work under the same assumptions of LapSVM using a similar mathematical machinery (as discussed in section 2) but optimize the squared loss instead. By having all the competitor methods optimizing the same loss, the effects observed come from the models only and not from the usage of different losses.

At last, since PNT-CMF performs dimensionality reduction and LapSVM operates on a lower dimensional manifold, we add a fourth competitor method which incorporates dimensionality reduction to TSVM as well: we applied PCA dimensionality reduction to all datasets and ran TSVM using the transformed data. This method is called PCA+TSVM. For each dataset we used the first k PCA dimensions, k being the same number of latent dimensions used by PNT-CMF. As a TSVM implementation we used *SVMLight*[2]. For LapSVM we used the implementation from [7], which is also available for download[3].

5.4 Model Selection and Reproducibility of the Experiments

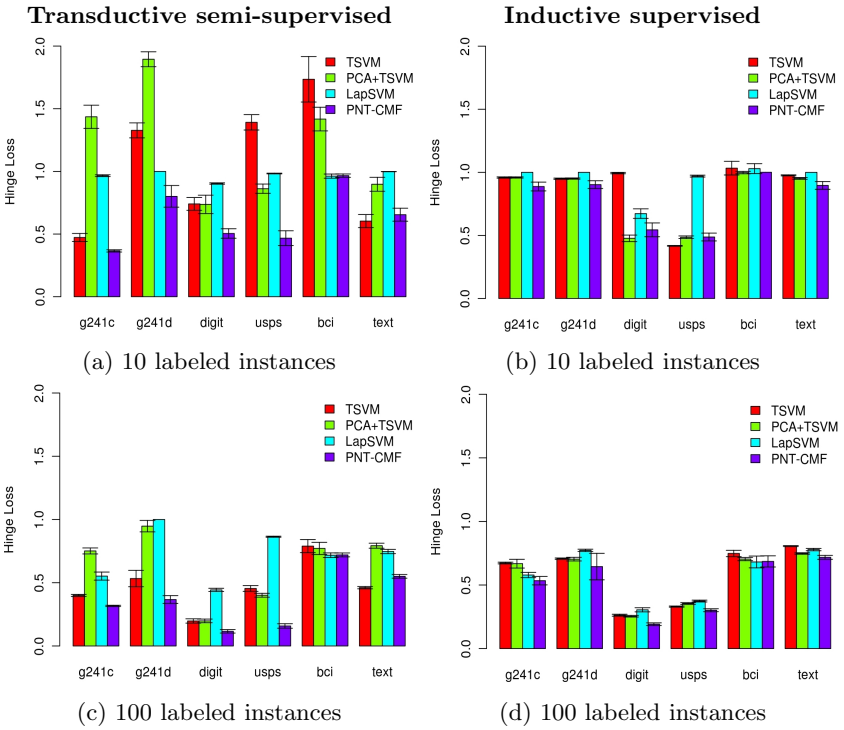
Model selection is a known problem for semi-supervised learning due to the low number of labeled instances available. We show that it is possible to estimate good hyperparameters for PNT-CMF even in the presence of only a few labeled data points. For **PNT-CMF**, each hyperparameter combination was evaluated through 5-fold cross-validation using **only the training data**. The code for PNT-CMF can be made available upon request to the first author. We gave the **baseline methods** a competitive advantage: use the same hyperparameter search approach but employing **both train and test data**. We also observe that the results for the competitor methods are consistent with the ones reported in the literature for the same datasets.

5.5 Results and Discussion

The hinge loss scores for the datasets with 10 and 100 labeled examples are shown in Figure 1. For each method and dataset, the average performance over the 12 splits is shown. The error bars represent the 99% confidence intervals. [2] divide

[2] `http://svmlight.joachims.org/`
[3] `http://www.dii.unisi.it/~melacci/lapsvmp/`

these datasets into two categories: the **manifold-like** and the **cluster-like**. The **manifold** group comprises the *digit*, *usps* and *bci* datasets in which the data lie near a low dimensional manifold. Algorithms like LapSVM are expected to excel in these datasets. *g241c*, *g241d* and *text* fall under the category of **cluster-like** datasets in which different classes do not share the same cluster thus making the optimal decision boundary to lie in a low density region, favoring algorithms like TSVM.



(a) 10 labeled instances (b) 10 labeled instances

(c) 100 labeled instances (d) 100 labeled instances

Fig. 1. Results for the Hinge Loss. The *lower* the better.

The left-hand side of Figure 1 shows how PNT-CMF performs in comparison to its competitors. By looking specially at this figure, one can see that TSVM is stronger in the *g241c* and *text* datasets while LapSVM is more competitive in the manifold-like data. One can also see that LapSVM is significantly weaker on the *g241c* and *text* sets where the data do not lie near a low dimensional manifold. PNT-CMF on the other hand is always either the statistically significant winner or is away from the winner by a non significant margin. Out of the 12 experiments, PNT-CMF is the sole winner in 8 of them and is one of the winning methods in all the 3 ties, with TSVM winning in one case, and only for 100 labeled instances. We show here the Hinge loss results because this was the measure all the models in the experiment are optimized for. As already said

we also evaluated the AUC of these models in these experiments and the results were similar, with PNT CMF being the sole winner in 5 experiments and one of the winning methods in the other 7. This supports our claim that PNT-CMF can consistently work well under different assumptions. In accordance with our expectations, LapSVM is more competitive on the manifold-like datasets while TSVM on the cluster-like ones.

Finally, by comparing the right and left hand sides of Figure 1, we can have an idea of the effects of taking unlabeled data into account. One can observe that TSVM seems to be more unstable, having sometimes worse hinge loss on the semi-supervised case than the corresponding ones on the supervised scenarios for the *bci*, *usps* and *g241d* datasets. The same happens for LapSVM on *g241d* dataset. PNT-CMF seems to be more robust in this sense, not presenting a significant performance degradation in any case.

6 Conclusion

In this work we proposed PNT-CMF, a factorization model for semi-supervised classification and showed how to learn such models in a transductive (semi-supervised) and in an inductive (supervised) setting. The performance of such models was evaluated on a number of different synthetic and real-world datasets with varying characteristics. The proposed model relies on the reconstruction of the predictors and the neighborhood matrices in the original feature space to learn latent factors used for classification. The contribution of each of them to the model can be controlled in order to fit different kinds of datasets better.

PNT-CMF represents a step forward in the state-of-the-art because, unlike other semi-supervised methods which face a performance degradation when their model assumptions do not hold, the experimental results showed that PNT-CMF is capable of coping with datasets with different characteristics. One evidence for this is that, in all cases, regardless of whether LapSVM or TSVM were the best models, PNT-CMF was always a strong competitor, being among the winners in the vast majority of the semi-supervised experiments.

As future work we plan to investigate better factorization strategies for the matrix K, like different loss and reconstruction functions. Also the extension and evaluation of the model on regression, multi-class and multi-label classification tasks is an issue to be further investigated.

Acknowledgments. The authors gratefully acknowledge the co-funding of their work by the Multi-relational Factorization Models project granted by the Deutsche Forschungsgesellschaft[4]. Lucas Drumond is sponsored by a scholarship from CNPq, a Brazilian government institution for scientific development.

[4] `http://www.ismll.uni-hildesheim.de/projekte/dfg_multirel_en.html`

References

1. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. The Journal of Machine Learning Research 7, 2399–2434 (2006)
2. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)
3. Cozman, F., Cohen, I., Cirelo, M.: Semi-supervised learning of mixture models. In: 20th International Conference on Machine Learning, vol. 20, pp. 99–106 (2003)
4. Gammerman, A., Vovk, V., Vapnik, V.: Learning by Transduction. In: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, pp. 148–156. Morgan Kaufmann (1998)
5. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In: Proceedings of the 1999 International Conference on Machine Learning, ICML (1999)
6. Liu, Y., Jin, R., Yang, L.: Semi-supervised multi-label learning by constrained non-negative matrix factorization. In: Proceedings of the National Conference on Artificial Intelligence, vol. 21, p. 421. AAAI Press (2006)
7. Melacci, S., Belkin, M.: Laplacian Support Vector Machines Trained in the Primal. Journal of Machine Learning Research 12, 1149–1184 (2011)
8. Nigam, K., McCallum, A., Mitchell, T.: Semi-supervised text classification using EM. In: Chapelle, O., Schölkopf, B., Zien, A. (eds.) Semi-Supervised Learning, pp. 33–56. The MIT Press, Cambridge (2006)
9. Rennie, J.: Smooth Hinge Classification (February 2005), `http://people.csail.mit.edu/jrennie/writing/smoothHinge.pdf`
10. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 650–658. ACM, New York (2008)
11. Szummer, M., Jaakkola, T.: Information regularization with partially labeled data. In: Advances in Neural Information Processing Systems, vol. 15, pp. 1025–1032 (2002)
12. Wang, F., Li, T., Zhang, C.: Semi-supervised clustering via matrix factorization. In: Proceedings of the 2008 SIAM International Conference on Data Mining, pp. 1–12. SIAM (2008)
13. Weinberger, K., Packer, B., Saul, L.: Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, pp. 381–388 (2005)
14. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Scholkopf, B.: Learning with local and global consistency. In: Advances in Neural Information Processing Systems, vol. 16, pp. 321–328 (2004)
15. Zhu, X.: Semi-supervised learning literature survey. Tech. Rep. 1530, University of Wisconsin, Madison (December 2006)
16. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML), pp. 912–919 (2003)