Online Matrix Factorization for Space Embedding Multilabel Annotation

Sebastian Otálora-Montenegro,
Santiago A. Pérez-Rubiano, and Fabio A. González

MindLab Research Group, Universidad Nacional de Colombia, Bogotá, Colombia
{jsotaloram,saaperezru,fagonzalezo}@unal.edu.co

Abstract. The paper presents an online matrix factorization algorithm for multilabel learning. This method addresses the multi-label annotation problem finding a joint embedding that represents both instances and labels in a common latent space. An important characteristic of the novel method is its scalability, which is a consequence of its formulation as an online learning algorithm. The method was systematically evaluated in different standard datasets and compared against state-of-the-art space embedding multi-label learning algorithms showing competitive results.

1 Introduction

The multilabel learning problem consists in inducing a function, from a set of labeled instances, that assigns one or more labels to a new instance. Formally, given a set of labels $\mathcal{L} = \{\lambda_1, \lambda_2, \ldots, \lambda_m\}$ and a set of instances $\mathcal{D} = \{x_i, y_i\}_{i=1\ldots n}$, where $x_i \in \mathcal{X}, y_i \subset \mathcal{L}$, with $l = |\mathcal{L}| > 1$, find a function $f : \mathcal{X} \to P(\mathcal{L})$.This is in contrast with the multi class learning problem where a given instance is associated with one and only one of the $|\mathcal{L}|$ labels. Instances are usually represented in a k-dimensional real space, i.e. $\mathcal{X} = R^k$, while the labels are usually represented using a l-dimensional vector space model, i.e. a subset of $\mathcal{L}$ is described by binary vector $y \in \{0, 1\}^l$ such that for $i = 1, \ldots, l$, $y_j[i] = 1$ if and only if the j-th instance has the i-th label associated with it.

The multi-label learning problem arises in areas such as semantic labeling of images and video, text classification, music categorization and functional genomics among others [9]. Several methods to address this problem have been proposed during the last years. Many of these methods transform the problem to a conventional classification problem. For instance, multiples classifiers are trained, one binary classifier per label, then a new instance is labeled by independently applying the set of classifiers. The problem with this class of approaches is that usually they do not scale well when there is a large number of labels and/or instances [9]. A possible strategy to deal with the large number of labels is to find a compact representation of them by using, for instance, a dimensionality reduction method. This approach is followed by multi-label latent space embedding (MLLSE) methods, which have recently shown competitive results [8,7,6].

J. Ruiz-Shulcloper and G. Sanniti di Baja (Eds.): CIARP 2013, Part I, LNCS 8258, pp. 343–350, 2013.

LSE methods find a low-dimensional latent space, $\mathcal{S} = R^s$ with $s << k$ or $s << l$, where labels, and in some case instances, are embedded. This embedding is expected to preserve and highlight correlation information from each space and to remove irrelevant, redundant or noisy data, while at the same time reducing computational and space complexities of the learning algorithms. The main disadvantage of this method is that usually the dimensionality of the shared space is a parameter to be tuned.

MLLSE methods use different strategies to find the latent space: principal component analysis, canonical correlation analysis, compressive sensing, among others. Some of these methods do not scale well to large datasets. This paper presents an MLLSE method, which is able to deal with large datasets thanks to the fact that it uses an online learning strategy. Our method embeds both the instances and the labels in a common latent space, which makes it able to exploit the labels structure and to find instance-instance, label-label and instance-label correlations.

The paper is organized as follows: Section 2 reviews the related work, Section 3 introduces the details of the new method, Section 4 presents the experimental evaluation of the method, and Section 5 discusses the conclusions as well as the future work.

2 Related Work

MLLSE models are characterized by the usage of a latent space where either the representation of the labels, the instances or both are embedded. This embedding is done to (1) reduce the dimensionality of the data and eventually reduce the computational and spatial complexities of the learning algorithms and to (2) extract and exploit hidden structures which rise from the correlation patterns present in real world datasets.

Among MLLSE models there is a subset that has been recently called feature-unaware [3]. These subset of methods only embed the labels' space but not the instances' space. Some examples of feature-unaware models are Compressive Sensing (CS)[4] and Principal Label Space Transformation (PLST)[8]. These models can be characterized by their definition of two functions: a label embedding function $f_L : \{0,1\}^l \to R^s$ and a reconstruction function $f_S : R^s \to \{0,1\}^l$. The function f_L goes from the original space of labels to the compact embedded representation and the function f_S reconstructs the original labels given the embedded representation. The labels embedding function is designed to exploit the correlation information and find semantic factors presents in the labels representation data, while the reconstruction function is designed to minimize the loss of information (as measured by a reconstruction error). Using the embedding, s linear regressors $f_i : \chi \to R$ are found and used later to predict the embedding of the labels of new unseen instances as $F(x) = (f_1(x), \ldots, f_s(x))$. This prediction is then translated into a labels representation using the reconstruction function.

In CS [4] the goal is to learn to predict compressed label vectors using linear regressors, under the assumption that the labels vector y are sparse. The embedding function is determined by random projections that allow the embedded labels to fulfill some sparseness requirements that reduce the number

of linear regressors to learn. The reconstruction function on the other hand is posed as an optimization problem, also called the decoding problem, and so adds computational complexity to the overall multi labeling process.

In PLST [8] the embedding function is defined as $f_L(y) = Vy$ and the recovery function is defined as $f_S(h) = V^T h$, where V is obtained by solving the optimization problem $\min_{VV^T=I} \|Z - V^T VZ\|$, with $Z = Y - \frac{1}{n} Y\mathbf{1}\mathbf{1}^T$, and $\mathbf{1}$ being a vector of ones. This minimizes the reconstruction error in the reconstruction function. In this model the reconstruction function consists in just a multiplication by a precomputed projector, in contrast with the decoding problem faced by CS.

There are two problems with feature-unaware models: (1) the information provided by the correlation between the instances' data and the labels' data is ignored as the dimensionality reduction is only applied to the labels' space, and (2) the curse of dimensionality is still present as the instances' space is not reduced. Reducing only the instances' space would have the same issues and so it is natural to think in models that embed both the labels and the instances into the same low-dimensional space.

In contrast with feature-unaware models, there are feature-aware models, which take into account both the instances and labels data when inducing the mappings to a latent space. Therefore an instances' embedding function $f_K : \mathcal{R}^k \to \mathcal{R}^s$ is learned in addition to the labels' embedding function and the reconstruction function. Examples of this models are multilabel max-margin embedding (MME) [6] and multilabel canonical correlation analysis (CCA) [7].

CCA [7] looks for a latent space in which the instances' and the labels' embedding correlation is maximized. The method proposed in [3] embeds the instances into the latent space to learn a binary relevance model in which a binary classifier is trained for each label, i.e. the labels' embeddings to the latent space are not further used and so there's no recovery function here.

In MME [6] both the instances and the labels are embedded into a shared latent space, where the distance between a given instance embedding and its associated labels' embedding is smaller than the distance between the instance embedding and other unrelated labels' embeddings in the dataset, i.e. $\forall j \neq i$ $f_K(x_i)^T f_L(y_i) - f_K(x_i)^T f_L(y_j)$ is maximized. Once such latent space is found MME predicts the labels of a new instance by embedding it to the latent space and then recovering the labels representation using the reconstruction function.

There are several MLLSE methods based on non-negative matrix factorization, those methods includes asymmetric and mixed NMF [2] and another variations like structure preserving NMF [5] that could be adapted to multilabel learning, however, they impose more constraints on the embedding space that usually leads to an addition of complexity to the original NMF model.

Our method simultaneously addresses both the embedding problem and the label representation reconstruction problem using a common formulation based on matrix factorization, which is solved using an efficient online learning strategy.

3 Online Matrix Factorization Multilabel Classification

Let $X \in R^{k \times n}$ be the instances representation matrix, with n the number of instances and k the dimensionality of the instances space, and $Y \in R^{l \times n}$ be the label indicator matrix with l the number of labels and $Y_{ij} = 1$ if and only if the i-th label is assigned to the j-th instance.

The proposed method has two stages, a learning stage and a prediction stage. In the learning stage the method assumes that there is a latent space, $L = \mathbb{R}^s$, where both instances and labels have a common representation. Lets $H \in \mathbb{R}^{s \times n}$ be the latent representation matrix, then X may be obtained from H by a transformation:

$$X = PH$$

In the same way, the label indicator matrix may be obtained from H by the transformation:

$$Y = QH$$

The goal of the learning stage is to find P, Q and H. This is accomplish by solving the following unconstrained optimization problem:

$$\min_{P,Q,H} (1-\alpha)\|X - PH\|_F^2 + \alpha\|Y - QH\|_F^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2 + \|H\|_F^2) \quad (1)$$

where α controls the relative importance of instance reconstruction with respect to label reconstruction during the embedding learning, and λ controls the relative importance of the regularization term, which penalizes large values (measured by the Frobenious norm) in matrices P and Q preventing overfitting.

The H matrix contains a compact representation of each instance, which is expected to encode a unique semantic representation of the instance, from which the label or instance information may be reconstructed using the Q and P transformations respectively.

In the prediction stage, the latent representation of a new unseen instance x is calculated using the the learned model (matrices P and Q) and solving the optimization problem:

$$h = \arg\min \|x - Ph\|_F^2 + \xi\|h\|_F^2 \quad (2)$$

From the latent representation h of the sample we calculate the corresponding labels using the Q matrix, i.e. the predicted labels representation for the new instance would be $y = Qh$. So, once the model is learned, the embedding function of our model consists in solving the convex optimization problem of Eq. 2, while the recovery function consists in multiplying by a projection matrix Q.

The algorithm 1 solves the optimization problem in Eq. 1 using an online learning strategy based on stochastic gradient descent. This requires the objective function to be formulated as a per-sample loss function as follows:

$$f(x_i, y_i, P, Q, h_i) = (1-\alpha)\|x_i - Ph_i\|_F^2 + \alpha\|y_i - Qh_i\|_F^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2) \quad (3)$$

In each step, the algorithm processes a sample (x_i, y_i) by calculating the gradient of f, ∇f, and moving the parameters P and Q in the opposite direction:

$$g_P(x_i, P, Q, h_i, \alpha, \lambda) = \nabla_P f(x_i, y_i, P, Q, h_i) = \lambda P - (1-\alpha)(x_i - Ph_i)h_i^T$$

$$g_Q(y_i, P, Q, h_i, \alpha, \lambda) = \nabla_Q f(x_i, y_i, P, Q, h_i) = \lambda Q - \alpha(y_i - Qh_i)h_i^T$$

Algorithm 1. Learning stage

Precondition: $X \in \mathcal{R}^{k \times n}$ and $Y \in \mathcal{R}^{l \times n}$
  function TRAIN($X, Y, s, \lambda, epochs$)
    $P^{(0)} \leftarrow$ RANDOM-MATRIX(k,s)
    $Q^{(0)} \leftarrow$ RANDOM-MATRIX(l,s)
    $h^{(0)} \leftarrow$ RANDOM-MATRIX(s,1)
    $gamma \leftarrow 1$
    for $j \leftarrow 1$ to $epochs$ **do**
      for $i \leftarrow 1$ to n **do**
        $x, y \leftarrow$ SAMPLE-WITHOUT-REPLACEMENT(X,Y)
        $P^{(i*j)} \leftarrow P^{(i*j)-1} - \gamma g_P(x, \lambda, P^{(i*j)-1}, Q^{(i*j)-1}, h^{(i*j)-1})$
        $Q^{(i*j)} \leftarrow Q^{(i*j)-1} - \gamma g_Q f(y, \lambda, P^{(i*j)-1}, Q^{(i*j)-1}, h^{(i*j)-1})$
        $h^{(i*j)} \leftarrow (\lambda I_r + {P^{(i*j)}}^T P^{(i*j)} + {Q^{(i*j)}}^T Q^{(i*j)})^{-1}(P^T x + Q^T y)$
        $\gamma \leftarrow \frac{\gamma}{1+i\gamma\lambda j}$
      end for
      RESTART-SAMPLING()
    end for
    return $P^{epochs \times n}, Q^{epochs \times n}$
  end function

In Algorithm 1 the details of the training algorithm are shown. The algorithm randomly samples pairs (instance,labels) from the dataset $\mathcal{D}$ and updates P and Q the gradient of per-instance loss function f (Eq. 3). The vector h is updated using the closed-form solution to the optimization problem with respect to h. At every update only one sample is processed, in contrast with batch gradient descent algorithms where the whole dataset is processed during each update, therefore the memory usage is proportional to the size of one sample and the size of the matrices P and Q. This makes the algorithm suitable for large-scale applications. The algorithm can be easily extended to process not just one sample but a small sampled subset of the dataset, a minibatch, on each iteration. This has shown good results on online factorization algorithms [1].

In this algorithm the learning rate γ is updated on each iteration to improve the convergence rate as it is usual in SGD algorithms. The *epochs* parameter indicates the number of times that each sample from the dataset will be processed. This parameter is analogous to the number of iterations of a batch gradient descent algorithm.

In Algorithm 2 the prediction algorithm is detailed. A new instance is embedded into the latent space by solving the problem in eq 2, to later recover its label representation. However the predicted representation is not necessarily a binary one because our model does not restricts it to be, but we can interpret this representation as a label ranking and then either selecting the top m ranked labels (letting m be close to the average number of labels of each instance in the training data set) or using a threshold value to decide for each label whether it is selected or not. For the experimental evaluation of our method we evaluate both approaches and selected the best one using cross-validation.

Algorithm 2. Prediction stage

function PREDICT(x, P, Q, xi, $param$)
  $h \leftarrow (\xi I_r + {P^{(i)}}^T P^{(i)})^{-1}(P^T x)$
  $y \leftarrow Qh$
  $L \leftarrow SELECT - TOP(param, y)$ ▷ Or THRESHOLD(param,y)
  return L
end function

The parameter $param$ might either be an integer for the top selection case or a real for the thresholding case. The $THRESHOLD(param, y)$ procedure returns a list of the indices of the labels with a rank value in y bigger than $param$. The procedure SELECT-TOP(m, y) will return the index of the elements of m largest elements of y, SAMPLE-WITHOUT-REPLACEMENT(X, Y) will sample a pair (instance, labels) from the dataset, represented by the pair (X, Y), and store in x the instance and in y the labels representation. Subsequent calls to SAMPLE-WITHOUT-REPLACEMENT will not return the same values as before, unless the RESTART-SAMPLING procedure is invoked.

4 Experimental Evaluation

The method was evaluated on 5 standard multilabel datasets (described in Table 1) distributed by the *mulan* framework authors [10]. Results were compared against 7 MLLSE algorithms: CCA, OVA, MME, CS, PLST, and two batch non-negative matrix factorization techniques: Assymetric Non-negative Matrix Factorization (ANMF) and Mixed Non-negative Matrix Factorization (MNMF) [2]. We used the implementation kindly provided by the authors of [6] to compare it with our algorithm on the larger Mediamill dataset.

We used the same experimental setup as in [6], i.e. a random 5-fold cross validation schema for all experiments. One important parameter of our algorithm is the weight parameter α, which controls the relative importance of instances and labels. This parameter was experimentally tuned. The weight α has been experimentally shown to have low values, giving more importance to the label data. This can be seen as a consequence of the semantic richness underlying the

Table 1. Datasets considered in our experimental setup. The label cardinality is the average number of labels per instance.

Dataset	Labels	Examples	Features	Label Cardinality
Medical	45	978	1449	1.245
Corel5k	374	5000	500	3.522
Bibtex	159	7395	1836	2.402
Scene	6	2407	294	1.074
MediaMill	101	43907	120	4.376

words used for annotation. In the reconstruction of the label vector for the test instances a range of thresholds were evaluated according to the range of values of the reconstructed vector, so we assign 1 to the label j of the instance x^n if $x^n_j > threshold$. In a similar way we choose the top-k labels for each test sample selecting the highest k label values in the reconstruction of the labels vector.

Table 2 reports the performance of each method in terms of the micro f-measure. In all the cases, the presented method shows a competitive performance, obtaining in three of the five datasets the best one. ANMF and MNMF are based on ideas similar to the ones that support our method, learning a joint embedding space for instances and labels. In ANMF a compact label space is learned as a semantic basis for the joint embedding space, and in MNMF a concatenation of the label and instance matrices is made to learn the embedding space, also the instance and label matrices are weighted according to a parameter α for weight the contribution to of each representation to the embedding space representation. Both algorithms have the drawback that are not suitable for large scale datasets, since them handle the whole matrices X and Y becoming unfeasible to store in RAM for huge datasets. In contrast, the proposed method scales very well to large datasets thanks to its formulation as an online learning algorithm.

Table 2. Performance of each method in terms of f-measure, in parentheses the embedding space dimension. Results in bold are the best ones for each dataset. Results for OVA, CCA, CS and PLST correspond to the ones reported in [6].

	Corel5k	**Scene**	**Bibtex**	**Medical**	**Mediamill**
OVA	0.112	0.617	0.372	0.732	—
CCA[7]	0.150	0.610	0.404	0.404	—
CS[4]	0.086 (50)	0.499 (6)	0.332 (50)	0.499(50)	—
PLST[8]	0.074 (50)	0.539 (6)	0.283 (50)	0.539 (50)	—
MME[6]	0.178 (50)	**0.698** (6)	0.403 (50)	0.808 (70)	0.199 (350)
ANMF[2]	0.210 (30)	0.678 (10)	0.297 (140)	0.679 (70)	0.496 (350)
MNMF[2]	0.240 (35)	0.697 (10)	0.376 (140)	0.690 (350)	**0.510** (350)
Our Method	**0.26337** (40)	0.691 (10)	**0.436** (140)	**0.896** (70)	0.503 (350)

5 Conclusions

We have presented a novel latent space embedding method for multi label annotation which learns a joint embedding space using an online matrix factorization formulation that can deal with large datasets. The method was compared against state-of-art MLLSE methods showing competitive results, especially on data sets with a large number of annotations. This might be due to the semantic richness of the information in the labels space. This is an aspect that we plan to explore in more detail in the future as well as to compare it against recent online methods for multi labeling in web-scale datasets.

Acknowledgments. This work was supported by Colciencias grant 566 "Jóvenes Investigadores 2012" and partially funded by the following projects: Colciencias "Anotación Automática y Recuperación por Contenido de Imágenes Radiológicas Usando Semántica Latente", "Diseño e implementación de un sistema de cómputo sobre recursos heterogéneos para la identificación de estructuras atmosféricas en predicción climatológica" and LACCIR "Multimodal Image Retrieval to Support Medical Case-Based Scientific Literature Search".

References

1. Caicedo, J.C., González, F.A.: Multimodal fusion for image retrieval using matrix factorization. In: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval, ICMR 2012, pp. 56:1–56:8. ACM, New York (2012)
2. Caicedo, J.C., González, F.A., Jaafar, B.-A., Olfa, N.: Multimodal Representation, Indexing, Automated Annotation and Retrieval of Image Collections via Non-negative Matrix Factorization. Neurocomputing 76(1) (2012)
3. Chen, Y.-N., Lin, H.-T.: Feature-aware label space dimension reduction for multi-label classification. In: Advances in Neural Information Processing Systems 25, pp. 1538–1546 (2012)
4. Hsu, D., Kakade, S.M., Langford, J., Zhang, T.: Multi-label prediction via compressed sensing. arXiv preprint arXiv:0902.1284 (2009)
5. Li, Z., Liu, J., Lu, H.: Structure preserving non-negative matrix factorization for dimensionality reduction. Computer Vision and Image Understanding 117(9), 1175–1189 (2013)
6. Park, S., Choi, S.: Max-margin embedding for multi-label learning. Pattern Recognition Letters (2012)
7. Sun, L., Ji, S., Ye, J.: Canonical correlation analysis for multilabel classification: a least-squares formulation, extensions, and analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(1), 194–200 (2011)
8. Tai, F., Lin, H.-T.: Multilabel classification with principal label space transformation. Neural Computation 24(9), 2508–2542 (2012)
9. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook, pp. 667–685. Springer (2010)
10. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: A java library for multi-label learning. Journal of Machine Learning Research 12, 2411–2414 (2011)