Predicting labels for dyadic data

Aditya Menon and Charles Elkan

UCSD

September 8, 2010

Overview of talk

- We propose a new problem, dyadic label prediction, and explain why it is an important extension of supervised learning
  - Within-network classification is shown to be a special case of this problem
- We discuss how we can learn supervised latent features to solve the label prediction problem
- We qualitatively compare different approaches to special cases of the problem that arose in different communities
- Experimental results highlight some challenges with the learning task

Outline

The dyadic prediction problem

- Supervised learning:

  Labelled examples $(x_i, y_i) \to$ Predict label on unseen example x'

- Dyadic prediction:

  Labelled dyads $((r_i, c_i), y_i) \to$ Predict label on unseen dyad (r', c')

- Labels are the outcomes of the interaction between two (possibly heterogeneous) entities, or dyads
  - **Example**: (User, Movie) dyads with a label denoting the rating (collaborative filtering)
  - **Example**: (User, User) dyads with a label denoting whether the two users know each other (link prediction)

Dyadic prediction as matrix completion

- We can cast dyadic prediction as a form of matrix completion
- Imagine we have a matrix $X \in \mathcal{X}^{m \times n}$, whose rows are indexed by r_i and columns by c_i
- The space $\mathcal{X} = \mathcal{X}' \cup \{\text{“?”}\}$
  - The entries with value “?” are missing
- The dyadic prediction problem is to predict the value of the missing entries
  - Thus, we henceforth call the r_i's row objects, and the c_i's column objects

Dyadic and link prediction

- A closely connected problem is link prediction in graphs
- Here, we have a graph where only some edges are observed, and wish to predict the presence/absence of all edges
- There is a two-way reduction between the problems
  - Link prediction is dyadic prediction on an adjacency matrix
  - Dyadic prediction is link prediction on a bipartite graph comprising nodes for the rows and columns
- This close connection between the problems lets us apply link prediction methods for dyadic prediction, and vice versa
  - Will be necessary when comparing methods later in the talk

Latent feature methods for dyadic prediction

- A popular strategy for dyadic prediction is learning latent features
- Simplest form: $X \approx UV^T$, where $U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{n \times k}$
  - $k \ll \min(m, n)$ is the number of latent features
- Learn U, V via optimization of the (nonconvex) objective

$$||X - UV^T||_O^2 + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_V}{2}||V||_F^2$$

where $|| \cdot ||_O^2$ denotes the Frobenius norm only over non-missing entries

  - Can be thought of as a form of regularized SVD

Outline

Dyadic label prediction

- An important extension of dyadic prediction is a problem we call dyadic label prediction
- The new problem is:

$$\left.\begin{array}{c}\text{Labelled dyads } ((r_i, c_i), y_i) \\ + \\ \text{Labelled objects } (r_i, y_i^r)\end{array}\right. \rightarrow \text{Predict label on unseen object } r'$$

  i.e. we want to predict labels for the individual row/column objects
  - Optionally, predict the labels for dyads too
  - Here, we attach the label to the row object only without loss of generality
- We'll allow $y_i^r \in \{0, 1\}^L$, i.e. we'll allow for multi-label prediction

Dyadic label prediction as matrix completion

- We can interpret the new problem as a form of matrix completion too
- The input is twofold: the standard dyadic prediction matrix $X \in \mathcal{X}^{m \times n}$, and an additional matrix $Y \in \mathcal{Y}^{m \times L}$
- Each column of Y corresponds to an individual tag for a global label
  - As before, we can decompose $\mathcal{Y} = \{0, 1\} \cup \{``?''\}$, where "?" denotes a missing entry
  - Observe that Y can have an arbitrary pattern of missing entries, unlike in standard supervised learning
- The goal is to fill in the missing entries in Y
  - Optionally fill in the missing entries of X, if any

Applications of dyadic label prediction

- Dyadic label prediction has important real-world applications
  - Determine whether users in a collaborative filtering matrix are likely to respond to an ad campaign
  - Score suspiciousness of users in a social network e.g. score of how likely to be a terrorist
  - Predict which strain of bacteria are likely to appear in a food processing plant [2]

Dyadic label prediction and supervised learning

- Dyadic label prediction can be seen as a generalization of standard supervised learning
- We are predicting labels for individual examples, but:
  - We may not have explicit features (side information) for the examples
  - We have implicit relationship information between our training examples via the X matrix
  - The relationship information may have missing data
  - We may optionally want to predict relationship information between examples also

Within-network classification as dyadic label prediction

- Consider a graph $G = (V, E)$, where some subset $V' \subseteq V$ of nodes have labels
- The task of predicting labels for all nodes in $V - V'$ is known as within network classification
- We see that this is an instance of dyadic label prediction
  - The X matrix is the adjacency matrix of the graph, the Y matrix consists of the labels of the nodes
- Is there any reason the dyadic label interpretation is useful?

Why is the dyadic interpretation useful?

- We can look at within-network classification through the lens of dyadic prediction, and borrow from developments in that literature
- We can let the edges E be partially observed, thus combining link prediction with label prediction
- Existing methods for dyadic prediction can be applied to within-network classification
  - Can use advantages of dyadic prediction methods such as ability to use side information
  - We'll be looking at a method that learns latent features
  - Literature for within-network classification typically uses quite different principles

Outline

A latent feature approach to dyadic label prediction

- If we were given features for the row objects, predicting the missing labels in Y can be solved via supervised learning
- In our setting, we assume we don't have such features
  - But we can learn them using the latent feature approach!
  - If we model $X \approx UV^T$, then we can think of U as being a feature representation for the row objects
- Thus given U, we can learn a weight matrix W via ridge regression, e.g.:

$$\min_W ||Y - UW^T||_F^2 + \frac{\lambda_W}{2}||W||_F^2$$

- If we assume there are no missing entries in X, and set $\lambda_W = 0$, this is essentially **SocDim** [3]

The SocDim approach

- SocDim is a method for within-network classification on an undirected graph G
- We learn latent features from the adjacency matrix X of G
  - One first computes the modularity of the adjacency matrix:

$$\mathcal{Q}(X) = X - \frac{1}{2|E|} d d^T$$

    where d is a vector of node degrees
  - The latent features are taken to be the eigenvectors of $\mathcal{Q}(X)$
- These features are then used by a standard supervised learning algorithm to predict the labels

A supervised latent feature approach

- The SocDim approach assumes the same latent features are predictive for the labels in X and in Y
- We can instead learn U to jointly model the data and label matrices, yielding supervised latent features:

$$\min_{U,V,W} ||X - UV^T||_F^2 + ||Y - UW^T||_F^2 + \frac{1}{2}(\lambda_U ||U||_F^2 + \lambda_V ||V||_F^2 + \lambda_W ||W||_F^2).$$

- This is equivalent to

$$\min_{U,V,W} ||[XY] - U[V;W]^T||_F^2 + \frac{1}{2}(\lambda_U ||U||_F^2 + \lambda_V ||V||_F^2 + \lambda_W ||W||_F^2)$$

- Thus we're treating the labels as new movies!
  - We refer to these as "label movies"
  - Is this reduction too simplistic?

Why not use the reduction?

- We assume that our real goal is predicting the labels: reconstructing X is a means to that end
- Consider weighting the "label movies" to reflect this, with a tradeoff parameter μ

$$\min_{U,V,W} ||X - UV^T||_F^2 + \mu ||Y - UW^T||_F^2 + \frac{1}{2}(\lambda_U ||U||_F^2 + \lambda_V ||V||_F^2 + \lambda_W ||W||_F^2)$$

- If we assume there are no missing entries in X, then this is essentially **supervised matrix factorization** (SMF) [4]
  - This method was designed for directed graphs, unlike SocDim

From SMF to dyadic prediction

- We can move from the SMF approach to one based on dyadic prediction
- In general, doing so induces a number of advantages
  - We can deal with missing data in X
  - We can allow for partially observed rows in Y i.e. arbitrary missing data pattern in Y
- We'll specifically use the **LFL** model [1], which has further advantages
  - We can use extra side-information about the row objects
  - Predicting well-calibrated probabilities over the label entries
  - Dealing with both nominal and ordinal labels

The LFL model for dyadic prediction

- Assume we have a finite set of entries in the input matrix X, say $1, \ldots, R$
- The latent feature log-linear or LFL model uses the probability model

$$p(X_{ij} = r | U, V) = \frac{\exp((U_i^r)^T V_j^r)}{\sum_{r'} \exp((U_i^{r'})^T V_j^{r'})}$$

- That is, we keep a set of latent features U, V for each outcome $r = 1 \ldots R$
- Since we have a model for $p(X_{ij} = r | U, V)$, we can output well-calibrated probabilities for the outcomes

Incorporating side-information

- While we can solve dyadic prediction just using row and column identifiers, it is important to exploit side-information when present
  - Sometimes, these extra features can be very predictive for the individual matrix entries
  - They are essential to solve cold start problems, where there are no existing observations for a row/column
- We can use this information with LFL by modifying the probability model
- If a_i and b_j denote covariates for the rows and columns respectively, then form the new model:

$$p(X_{ij} = r | U, V) \propto \exp((U_i^r)^T V_j^r) + (w^r)^T \begin{bmatrix} a_i & b_j \end{bmatrix}).$$

Optimizing in the LFL model

- We can optimize either MSE or log-likelihood, depending on whether the entries are ordinal or not
    - This also affects our model's prediction
    - For nominal outcomes, we predict argmax $p(r|U, V)$
    - For ordinal outcomes, we predict $\sum_r rp(r|U, V)$

A general LFL model for graphs

- To allow for modelling graphs, we can consider the following generalization of the LFL model:

$$p(X_{ij} = r | U, V, \Lambda) \propto e^{(U_i^r)^T \Lambda_{ij} V_j^r}.$$

- We can constrain the latent features depending on the nature of the input
  - For general dyadic data where the rows and columns are distinct graphs, we let $\Lambda = I$
  - For asymmetric graph data, we set $V = U$ but let Λ be an arbitrary dense matrix [4]
  - For symmetric graph data, we set $V = U$ and $\Lambda = I$

Using the LFL model for label prediction

- Assuming the entries in X have ordinal structure, the joint optimization is:

$$\min_{U,V,W} ||X - \mathcal{E}(X)||^2_{\mathcal{O}} + \frac{1}{2}\left(\sum_r \lambda_U ||U^r||^2_F + \lambda_V ||V^r||^2_F\right) + \sum_{(i,l)\in\mathcal{O}} \frac{e^{Y_{il}(W_l^T U_i)}}{1 + e^{W_l^T U_i}} + \frac{\lambda_W}{2}||W||^2_F$$

  where

$$\mathcal{E}(X)_{ij} = \sum_r r \cdot p(X_{ij} = r | U, V)$$

- Thus, we fill in the missing entries in X plus the missing labels in Y

Outline

An overview of approaches

- We've seen three different approaches to the label prediction problem
  - SocDim
  - SMF
  - LFL
- They haven't been compared before, although we now see that they naturally target the same problem
- How are they different to each other?

Comparison of approaches

- A summary of the differences between the methods:

Item	SocDim	SMF	LFL
Supervised latent features?	No	Yes	Yes
Asymmetric graphs?	No	Yes	Yes
Handles missing data in X?	No	No	Yes
Finds latent features of?	Modularity	Data	Data
Single minimum?	Yes	No	No

- We now look more closely at how some differences arise as a result of the objective function being optimized

Comparison of approaches: objective functions

- Let's compare the objective functions of the methods for a special case
- Since SocDim and SMF operate natively on graphs, we assume our input is a graph, with the following constraints:
  - Assume there is no missing data in X, for fairness to SocDim and SMF
  - Assume that the graph is undirected, since otherwise SocDim doesn't work
- Finally, assume that we don't learn latent features in a supervised manner, for fairness to SocDim

Comparison of approaches: objective functions

- **SocDim**: if $\mathcal{Q}$ denotes the modularity matrix, then

$$\min_{U,\Lambda \text{ diagonal}} ||\mathcal{Q}(X) - U\Lambda U^T||_F^2$$

- **Supervised matrix factorization**:

$$\min_{U,\Lambda} ||X - U\Lambda U^T||_F^2 + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_\Lambda}{2}||\Lambda||_F^2$$

- **LFL**: denoting $\sigma(x) = 1/(1+e^{-x})$,

$$\min_{U} ||X - \sigma(UU^T)||_F^2 + \frac{\lambda_U}{2}||U||_F^2$$

- Thus, in general:

$$\min_{U,\Lambda} ||f(X) - g(U,\Lambda)||_F^2 + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_\Lambda}{2}||\Lambda||_F^2$$

SocDim vs LFL

- SocDim and LFL transform different things: the data matrix and the estimate respectively
- Transforming the estimate ensures $[0, 1]$ predictions
- Transforming the input has analogues in spectral clustering
  - There, one uses the graph Laplacian, which attempts to normalize nodes wrt their degrees
  - There is some similarity to weighting the regularizer by the number of ratings
- It is unclear if the particular choice of transformation makes a difference
  - Does SocDim perform similarly if we use the Laplacian instead of modularity?

SocDim vs SMF

- These methods are essentially the same when we don't have supervised features nor missing data, but for two points:
  - SocDim operates on the modularity matrix, while SMF works on the raw data matrix
  - SocDim admits a closed form solution, while SMF does not; thus SocDim is immune to local optima
- The immunity to local optima potentially offsets the fact that SocDim is an unsupervised method

Outline

Relevant questions for empirical study

- Do supervised latent features help?
  - SocDim has the advantage of being immune to local optima
  - Does that counteract its use of unsupervised latent features?
- Which data transform is the best? Does it matter?
  - Can we use the Laplacian matrix for SocDim with equivalent performance?
  - Does using the modularity/Laplacian matrix for SMF improve performance?
- Can we get away with naïve approaches to dealing with missing edges?
  - For example, can we just use row/column averages of entries?
  - If so, then SocDim/SMF can be applied to a wider range of problems

Datasets used

- `blogcatalog`: Fully observed links between 2500 bloggers in the BlogCatalog directory. Labels are the users' interests, which are divided into 39 categories (multilabel problem)
- `senator`: "Yea"/"Nay" votes of 101 senators concerning 315 bills from the 109th house in the Senate. Label is whether or not a senator is a Republican or Democrat
- `usps`: Handwritten digits represented as grayscale 16×16 images. We occlude some pixels, so that X has missing entries. Labels are the true digits that the images correspond to
    - Shows how dyadic label prediction can solve a more difficult version of a supervised learning task

Error measures

- For binary prediction tasks (`senator` and `usps`), we report 0-1 error
- For multi-label tasks (`blogcatalog`), we report F1-micro and F1-macro scores
  - For true tags y_{il} and predictions $\hat{y}_{il}$, these are

$$\text{Micro} = 2\frac{\sum_{i,l} y_{il}\hat{y}_{il}}{\sum_{i,l} y_{il} + \hat{y}_{il}}$$

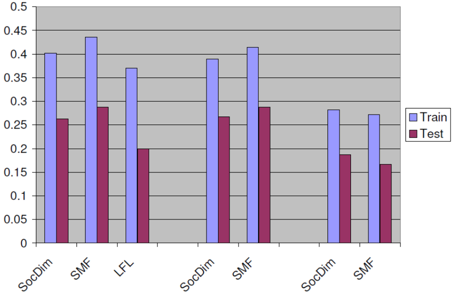
and

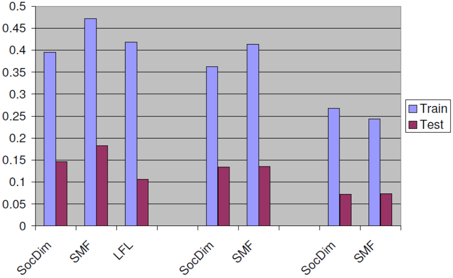
$$\text{Macro} = \frac{2}{L}\sum_{l}\frac{\sum_{i} y_{il}\hat{y}_{il}}{\sum_{i} y_{il} + \hat{y}_{il}}$$

- All errors are the result of 10-fold cross-validation

Results on blogcatalog

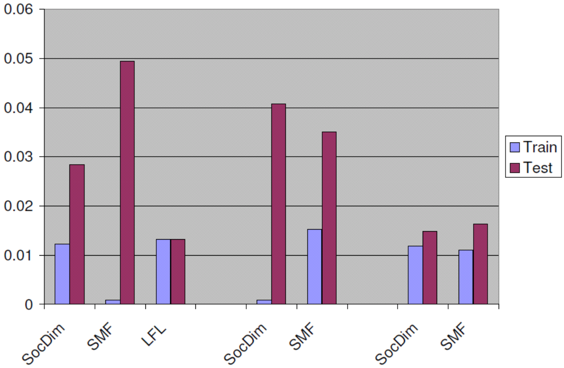
- SMF is the best performing method; the relatively poor performance of LFL suggests that the sigmoidal transform is not useful for this problem
- We find using the raw data matrix is sufficient to be as good as the modularity/Laplacian transform
- In general, the methods seem to overfit on the training data, despite the use of ℓ_2 regularization





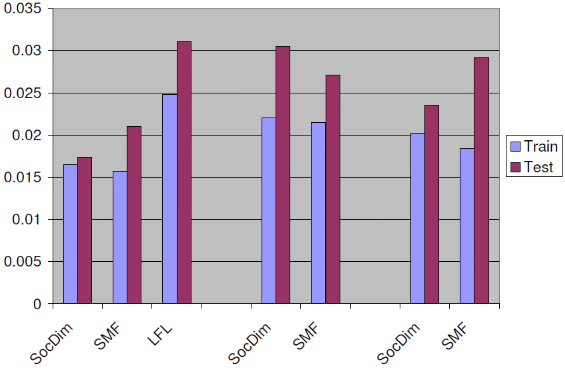
Results on senator

- LFL does best on this dataset; the other two methods sometimes overfit, achieving perfect training error!
- The Laplacian appears to offer marginal improvement as a data transformation



Results on usps

- SocDim surprisingly manages to do the best, despite simply ignoring the missing values
- Again, the raw data matrix gives the best results in general



Outline

Conclusion

- We proposed a new problem, dyadic label prediction, as an extension of both supervised learning and dyadic prediction
- We showed the relationship between the new problem and within-network classification
- We showed how we can use supervised latent features to predict the labels
- A comparison of different approaches to the problem reveals interesting issues in designing a solution to the problem
- Experimental results indicate there are issues with supervised latent features, requiring further research

Outline

References

Aditya Krishna Menon and Charles Elkan.
Dyadic prediction using a latent feature log-linear model.
http://arxiv.org/abs/1006.2156, 2010.

Purnamrita Sarkar, Lujie Chen, and Artur Dubrawski.
Dynamic network model for predicting occurrences of salmonella at food facilities.
In *Proceedings of the BioSecure International Workshop*, pages 56–63. Springer, 2008.

Lei Tang and Huan Liu.
Relational learning via latent social dimensions.
In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 817–826. ACM, 2009.

Shenghuo Zhu, Kai Yu, Yun Chi, and Yihong Gong.
Combining content and link for classification using matrix factorization.
In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 487–494. ACM, 2007.