The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Beyond Link Prediction: Predicting Hyperlinks in Adjacency Space

Muhan Zhang, Zhicheng Cui, Shali Jiang, Yixin Chen
Department of Computer Science and Engineering, Washington University in St. Louis
{muhan, z.cui, jiang.s}@wustl.edu, chen@cse.wustl.edu

Abstract

This paper addresses the hyperlink prediction problem in hypernetworks. Different from the traditional link prediction problem where only pairwise relations are considered as links, our task here is to predict the linkage of multiple nodes, i.e., hyperlink. Each hyperlink is a set of an arbitrary number of nodes which together form a multiway relationship. Hyperlink prediction is challenging – since the cardinality of a hyperlink is variable, existing classifiers based on a fixed number of input features become infeasible. Heuristic methods, such as the common neighbors and Katz index, do not work for hyperlink prediction, since they are restricted to pairwise similarities. In this paper, we formally define the hyperlink prediction problem, and propose a new algorithm called Coordinated Matrix Minimization (CMM), which alternately performs nonnegative matrix factorization and least square matching in the vertex adjacency space of the hypernetwork, in order to infer a subset of candidate hyperlinks that are most suitable to fill the training hypernetwork. We evaluate CMM on two novel tasks: predicting recipes of Chinese food, and finding missing reactions of metabolic networks. Experimental results demonstrate the superior performance of our method over many seemingly promising baselines.

Introduction

Link prediction (Liben-Nowell and Kleinberg 2007; Lü and Zhou 2011) has been studied broadly in recent years (Chen et al. 2015; Song, Meyer, and Tao 2015; Wu et al. 2016; Zhang and Chen 2017). Existing methods can be grouped into two types: topological feature-based approaches and latent feature-based approaches. Popular approaches include heuristic methods based on common neighbors, Jaccard coefficient, Katz index etc. (Liben-Nowell and Kleinberg 2007), and latent feature models (Miller, Jordan, and Griffiths 2009; Menon and Elkan 2011). These approaches, however, are restricted to predicting pairwise relations. None of them is directly applicable to predicting hyperlinks. A hyperlink relaxes the restriction that only two nodes can form a link. Instead, an arbitrary number of nodes are allowed to jointly form a hyperlink. A network made up of hyperlinks is called a hypernetwork or hypergraph.

Hypernetworks exist everywhere in our life. Examples include metabolic networks and citation networks. In metabolic networks, each reaction can be regarded as a hyperlink among its component metabolites. In citation networks, a hyperlink is a paper connecting all its authors. Due to the ability to model higher-order interactions between objects, hypernetworks have gained more and more popularity in application domains such as electronics (Karypis et al. 1999), finance (Bautu et al. 2009), and bioinformatics (Oyetunde et al. 2016).

Despite the popularity and importance of hypernetworks, there is still limited research on hyperlink prediction, i.e., to predict if a set of nodes is likely to be a hyperlink. One great challenge lies in the variable cardinality of hyperlinks. Existing supervised link prediction models are based on a fixed number of input features (features of the two target vertices). However, the number of vertices in a hyperlink is variable, making existing methods infeasible. On the other hand, link prediction methods based on topological features, such as common neighbors, cannot be applied to hyperlink prediction either, since these measures are defined for pairs of nodes instead of hyperlinks. As we will see in our experiments, a few naive generalizations of these measures have poor performance.

The variable cardinality problem not only prevents us from using traditional link prediction techniques, but also results in much larger inference space for hyperlink prediction. For a network with n vertices, the total number of potential **links** is only $\mathcal{O}(n^2)$. As a regular procedure in link prediction, we can list all the potential links and compute a score for each one. The ones with the highest scores are selected as predicted links. However, in hyperlink prediction, for the same network, the total number of potential **hyperlinks** is $\mathcal{O}(2^n)$. The exponential number of potential hyperlinks makes it impractical to list all the hyperlinks and give a score to each one of them.

Fortunately, in most cases we do not need to really consider all the potential hyperlinks, as most of them can be easily filtered out in particular problem settings. For example, in the task of finding missing metabolic reactions, we do not need to consider all 2^n possible reactions since most of them do not contain biological meanings. Instead, we can restrict the candidate hyperlinks to be the set of all actually feasible reactions. Also, in some problems, people may be interested only in hyperlinks with cardinalities less than a small number. For instance, in citation networks of computer science,

papers rarely have more than 10 authors. In such cases, the candidate hyperlinks are limited instead of exponential, and hyperlink prediction becomes a feasible problem.

Here, we formally define the hyperlink prediction problem. Let $H = \langle V, E\rangle$ be an incomplete hypernetwork, where $V = \{v_1, \ldots, v_n\}$ is the set of n vertices, and $E = \{e_1, \ldots, e_m\}$ is the set of m observed hyperlinks with each e_i being a subset of vertices in V. We assume some hyperlinks are missing from H. We use D to denote a set of candidate hyperlinks where we assume all the missing hyperlinks are contained in D.

Problem 1. *(Hyperlink Prediction) A hyperlink prediction problem is a tuple (H, D), where $H = \langle V, E\rangle$ is a given incomplete hypernetwork, and D is a set of candidate hyperlinks. The task is to find, among all hyperlinks in D, the most likely hyperlinks that are missing from H.*

A hypernetwork H can be conveniently represented as an *incidence matrix* $S \in \{0,1\}^{n\times m}$, where each column of S represents a hyperlink and each row represents a vertex. We use $[\cdot]_{ij}$ to denote the (i^{th} row, j^{th} column) of a matrix. We have: $S_{ij} = 1$ if $v_i \in e_j$; $S_{ij} = 0$ otherwise. Since S is incomplete, we let the missing hyperlinks be ΔS (also an incidence matrix, but unknown). We use an $n \times m'$ matrix U to denote the incidence matrix of D, where $m' = |D|$ is the number of candidate hyperlinks. Then, the hyperlink prediction problem becomes finding as many columns of ΔS as possible from U.

There are several seemingly promising baselines for hyperlink prediction. For instance, we may directly train a classifier on columns of S (with random negative sampling) and use it to classify U. However, our experiments show that such an approach has only slightly better performance than random guess. The reason is that hypernetworks are often extremely sparse, i.e., the number of observed hyperlinks m is far less than 2^n, which leads to a poor generalization ability. Another approach is to view hyperlink prediction as an information retrieval (IR) problem and use IR algorithms to retrieve hyperlinks from U according to query S. As we will show later, such an approach also has poor performance. This is because IR aims at finding items similar to the query instead of predicting unseen hyperlink relations.

The above observations suggest that it is inappropriate to model hyperlink prediction as a standard classification or IR problem. This implies the need to develop novel relationship modeling methods. However, directly modeling high-order relationships in the incidence space suffers from the variable cardinality problem, which prevents us from using existing link prediction techniques. In this paper, we propose to predict hyperlinks in the adjacency space. Our key observation is that a hyperlink $\mathbf{s}$ (a column vector in an incidence matrix) can be transformed into its equivalent matrix representation in the vertex adjacency space by $\mathbf{s}\mathbf{s}^\top$. This observation motivates us to first infer the pairwise relationships in the adjacency space leveraging existing link prediction techniques, and then find the missing hyperlinks through constrained optimization. Based upon this, we propose a two-step EM-style optimization method, Coordinated Matrix Minimization (CMM), which alternately performs nonnegative matrix factorization and least square matching to find a set of hyperlinks best suiting the given hypernetwork. We compare CMM with extensive baseline methods on predicting recipes and finding missing metabolic reactions, and demonstrate that our algorithm is currently the best hyperlink prediction algorithm for the considered tasks.

Coordinated Matrix Minimization

Since direct inference in incidence space is difficult, we choose to project hyperlinks into their vertex adjacency space and model hyperlinks in the adjacency space.

Given an incomplete hypernetwork S, we can calculate its adjacency matrix representation by $A = SS^\top$, where A_{ij} is the cooccurrence count of vertex i and j in all hyperlinks[1]. Since S is incomplete (some columns ΔS are missing), the resulting A is also incomplete.

Let the complete incidence matrix be $[S, \Delta S]$, where we use $[\cdot,\cdot]$ to denote horizontal concatenation. We can calculate its adjacency matrix as follows:

$$\begin{aligned}[S, \Delta S][S, \Delta S]^\top &= SS^\top + \Delta S \Delta S^\top \\ &= A + \Delta A, \end{aligned} \tag{1}$$

where we define $\Delta A = \Delta S \Delta S^\top$. We notice that the adjacency matrix A is also subjected to a loss ΔA. The columns of ΔS are missing from S and are in the candidate incidence matrix U. Our task is to find out these missing columns.

For convenience, we write $U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{m'}]$, where $\mathbf{u}_i$ is the i^{th} column of U. Let a diagonal matrix $\Lambda = \text{diag}([\lambda_1, \ldots, \lambda_{m'}])$ be an indicator matrix for columns of U, where $\lambda_i = 1$ indicates that hyperlink $\mathbf{u}_i$ is a column in ΔS and $\lambda_i = 0$ otherwise. Then, assuming Λ is known, the loss ΔA can be expressed as:

$$\Delta A = U \Lambda U^\top. \tag{2}$$

To model the (nonnegative) complete adjacency matrix $A + U\Lambda U^\top$, we adopt a nonnegative matrix factorization framework. Let an $n \times k$ nonnegative matrix $W = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n]^\top$ be the latent factor matrix, where $\mathbf{w}_i^\top$ is a row vector containing k latent features of vertex i ($k \ll n$). We assume the complete adjacency matrix is factored by

$$A + U\Lambda U^\top \approx WW^\top, \tag{3}$$

subject to some noise. To find the missing hyperlinks, we propose the following optimization problem:

$$\begin{aligned} \underset{\Lambda, W}{\text{minimize}} \quad & \left\| A + U\Lambda U^\top - WW^\top \right\|_{\text{F}}^2, \\ \text{subject to} \quad & \lambda_i \in \{0,1\},\ i = 1, \ldots, m' \\ & W \geq 0. \end{aligned} \tag{4}$$

[1] In general, A_{ij} can represent a weighted count if we consider hyperlink weights. Let $V = \text{diag}([v_1, \ldots, v_m])$ be a real nonnegative weight matrix. The weighted adjacency matrix of S becomes $A = SVS^\top$, where A becomes a real matrix. In this paper, we assume $V = I$, although weights can be handled as well.

Intuitively, we aim to simultaneously find a subset of candidate hyperlinks (given by Λ) as well as a latent factor matrix W that best explains the complete adjacency matrix $A+U\Lambda U^\top$. The proposed problem (4) also has a nice EM formulation, which naturally leads to a two-step alternate optimization algorithm. We explain it in the following.

EM formulation

We use Gaussian distribution to model the noise in (3) and define the conditional distribution of $A+U\Lambda U^\top$ as

$$p\big(A+U\Lambda U^\top \mid \Lambda, W, \sigma^2\big) = \prod_{i=1}^{n}\prod_{j=1}^{n}\mathcal{N}\big([A+U\Lambda U^\top]_{ij} \mid \mathbf{w}_i{}^\top \mathbf{w}_j, \sigma^2\big). \quad (5)$$

Consequently, we have the conditional distribution of the observed adjacency matrix A:

$$p\big(A \mid \Lambda, W, \sigma^2\big) = \prod_{i=1}^{n}\prod_{j=1}^{n}\mathcal{N}\big(A_{ij} \mid \mathbf{w}_i{}^\top \mathbf{w}_j - [U\Lambda U^\top]_{ij}, \sigma^2\big). \quad (6)$$

We also assume that each binary λ_i in Λ has an independent Bernoulli distribution:

$$p(\Lambda \mid \theta) = \prod_{i=1}^{m'}\theta^{\lambda_i}(1-\theta)^{1-\lambda_i}. \quad (7)$$

Now, the marginal distribution of A is

$$p\big(A \mid W, \sigma^2, \theta\big) = \sum_{\Lambda} p\big(A \mid \Lambda, W, \sigma^2\big)p(\Lambda \mid \theta). \quad (8)$$

We use maximum likelihood to estimate the parameters, the goal of which is to maximize the likelihood function of the observed data A given by (8). The hidden variable Λ inside the summation reminds us of the Expectation-Maximization (EM) algorithm (Dempster, Laird, and Rubin 1977). Let $\Theta = (W, \sigma^2, \theta)$ be the collection of all parameters. The E-step involves calculating the expectation of the complete data log-likelihood $\ln p(A, \Lambda \mid \Theta)$ w.r.t. the posterior distribution of Λ given the old parameter estimates. The posterior distribution of Λ is given by

$$p\big(\Lambda \mid A, \Theta^{\text{old}}\big) = \frac{p\big(A \mid \Lambda, \Theta^{\text{old}}\big)p\big(\Lambda \mid \Theta^{\text{old}}\big)}{\sum_{\Lambda'} p\big(A \mid \Lambda', \Theta^{\text{old}}\big)p\big(\Lambda' \mid \Theta^{\text{old}}\big)} = \frac{\exp\{-\big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2/2\sigma^2\}\prod_{i=1}^{m'}\theta^{\lambda_i}(1-\theta)^{1-\lambda_i}}{\sum_{\Lambda'}\exp\{-\|A-WW^\top+U\Lambda' U^\top\|_{\text{F}}^2/2\sigma^2\}\prod_{i=1}^{m'}\theta^{\lambda'_i}(1-\theta)^{1-\lambda'_i}}. \quad (9)$$

And the expectation of the complete data log-likelihood which we aim to maximize is

$$\mathcal{Q}(\Theta) = \sum_{\Lambda} p\big(\Lambda \mid A, \Theta^{\text{old}}\big)\ln p(A, \Lambda \mid \Theta), \quad (10)$$

where

$$\ln p(A, \Lambda \mid \Theta) = \ln p\big(A \mid \Lambda, W, \sigma^2\big) + \ln p(\Lambda \mid \theta) = \sum_{i=1}^{n}\sum_{j=1}^{n}\big[-\frac{1}{2}\ln 2\pi\sigma^2 - \frac{1}{2\sigma^2}(A_{ij} - \mathbf{w}_i{}^\top\mathbf{w}_j + [U\Lambda U^\top]_{ij})^2\big] + \sum_{i=1}^{m'}[\lambda_i \ln\theta + (1-\lambda_i)\ln(1-\theta)]. \quad (11)$$

The difficulty in maximizing (10) is that the posterior distribution (9) of Λ does not factorize over its m' components, thus evaluating (10) requires the summation over all $2^{m'}$ possible states of Λ, leading to prohibitively expensive calculations.

To achieve a simple and elegant approximate solution, we resort to a hard indicator matrix Λ. Consider the posterior distribution of Λ given by (9). Assume the variance $\sigma^2 \to 0$, and assume $\theta \in (0, 1)$. Then, both the numerator and the denominator will go to zero. However, in the denominator, the term with the smallest $\big\|A-WW^\top+U\Lambda' U^\top\big\|_{\text{F}}^2$ will go to zero most slowly. This means that $p\big(\Lambda|A, \Theta^{\text{old}}\big)$ will be zero for all Λ except for $\arg\min_\Lambda \big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2$, whose probability will go to 1. Therefore, we obtain a hard indicator Λ with all the posterior distribution centered at one point.

Our **E-step** becomes, under fixed W,

$$\begin{aligned} \underset{\Lambda}{\text{minimize}} \quad & \big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2, \\ \text{subject to} \quad & \lambda_i \in \{0, 1\},\ i = 1, \ldots, m', \end{aligned} \quad (12)$$

We still use Λ to denote the minimum optimized from (12). After getting Λ, (10) reduces to

$$\mathcal{Q}(\Theta) = \ln p(A, \Lambda \mid \Theta), \quad (13)$$

where the complete data log-likelihood is given by (11).

The M-step is maximizing $\mathcal{Q}(\Theta)$ to update the parameter estimates. Setting the derivative w.r.t. θ in (11) to be zero, we obtain $\theta = (\sum_{i=1}^{m'}\lambda_i)/m'$. Under reasonable initializations, θ will always be within $(0, 1)$. Since σ is an (infinitesimally small) constant, we can optimize W independently of θ and σ, leaving us with the objective function

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(A_{ij} - \mathbf{w}_i{}^\top\mathbf{w}_j + [U\Lambda U^\top]_{ij})^2 = \big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2. \quad (14)$$

Therefore, our **M-step** becomes, under fixed Λ,

$$\begin{aligned} \underset{W}{\text{minimize}} \quad & \big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2, \\ \text{subject to} \quad & W \geq 0, \end{aligned} \quad (15)$$

As we can see, by assuming $\sigma^2 \to 0$ we obtain a simple two-step optimization procedure with a single objective function $\big\|A-WW^\top+U\Lambda U^\top\big\|_{\text{F}}^2$. The E-step optimizes Λ with W fixed, while the M-step optimizes W with Λ fixed.

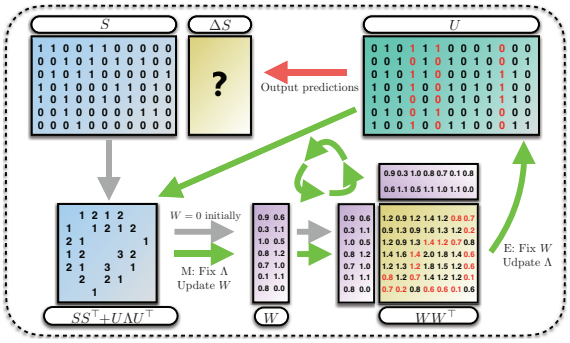


Figure 1: An illustration of CMM. The incomplete incidence matrix S is first transformed into its adjacency matrix. The M-step optimizes W with Λ fixed. The E-step optimizes Λ with W fixed. This procedure is iterated until convergence.

Thus, the EM steps exactly correspond to an alternate optimization over the two matrices Λ and W. We call the resulting algorithm Coordinated Matrix Minimization (CMM), which is shown in Algorithm 1. Since each of the two steps decrease the objective function, CMM is guaranteed to converge to a local minimum. We illustrate CMM in Figure 1.

Solving individual EM steps

Now we discuss how to solve the individual E and M steps.

For the E-step given by (12), we first show that it can be transformed to an integer least square problem. Note that

$$U\Lambda U^\top = \sum_{i=1}^{m'} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top. \tag{16}$$

We reshape the $n \times n$ matrix $\mathbf{u}_i\mathbf{u}_i^\top$ into an $n^2 \times 1$ vector $\mathbf{c}_i$ by vertically concatenating its columns, and let $C = [\mathbf{c}_1, \ldots, \mathbf{c}_{m'}]$. We also reshape the $n \times n$ matrix $A - WW^\top$ into an $n^2 \times 1$ vector $-\mathbf{d}$, and use $\mathbf{x}$ to denote the vector $[\lambda_1, \ldots, \lambda_{m'}]^\top$. Then, we can transform (12) into the following form:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \|C\mathbf{x} - \mathbf{d}\|_2^2, \\ \text{subject to} \quad & \mathbf{x} \in \{0,1\}^{m'}, \end{aligned} \tag{17}$$

which is a standard integer least square form.

We know that integer least square problem is NP-hard. When m' is large, it is generally intractable. Therefore, we follow a regular procedure to relax the constraint of λ_i to be continuous within $[0, 1]$. The optimization problem becomes a constrained linear least square problem, which can be solved very efficiently using off-the-shelf optimization tools. These continuous scores λ_is can be viewed as soft indicators of the candidate hyperlinks. Note that in order to ensure convergence, we do not round Λ after each iteration, but consistently optimize over the continuous Λ.

For the M-step given by (15), it is a symmetric nonnegative matrix factorization problem. We use an improved projected Newton algorithm proposed by (Kuang, Ding, and Park 2012). More specifically, the iterative update rule is:

$$\mathbf{x}_{\text{new}} = [\mathbf{x} - \alpha \mathbf{H}^{-1} \nabla f(\mathbf{x})]^+, \tag{18}$$

where $\mathbf{x}$ is the vectorized W, f is the objective function in (15), $\mathbf{H}^{-1}$ is a modified inverse Hessian matrix of $f(\mathbf{x})$, α is the step size, and $[\cdot]^+$ denotes the projection to the nonnegative orthant. The gradient $\nabla f(\mathbf{x})$ has an analytical form of $\text{vec}\big(4(WW^\top - A - U\Lambda U^\top)W\big)$. It is shown that with some mild restrictions on $\mathbf{H}^{-1}$, the iterative algorithm is guaranteed to converge to a stationary point.

Our CMM algorithm iteratively performs the two steps until a convergence threshold is satisfied or a maximum iteration number is reached. We use the final scores Λ to rank all candidate hyperlinks and select the top ones as predictions.

Algorithm 1 Coordinated Matrix Minimization

1: **input:** Observed hyperlinks S, candidate hyperlinks U.
2: **output:** Indicator matrix Λ.
3: Calculate $A = SS^\top$. Initialize W and Λ to zero.
4: **while** Λ has not converged **do**
5: E-step: solve (12).
6: M-step: solve (15).
7: **end while**
8: Select candidate hyperlinks according to Λ.

Related Work

Although hyperlinks are common in real world and can be used to model multiway relationships, currently there are still limited research on hyperlink prediction. Xu et al. (Xu, Rockmore, and Kleinbaum 2013) proposed a supervised HPLSF framework to predict hyperlinks in social networks. To deal with the variable number of features, HPLSF uses their entropy score as a fixed-length feature for training a classification model. To the best of our knowledge, this is the only algorithm that is specifically designed for hyperlink prediction in arbitrary-cardinality hypernetworks.

Nevertheless, learning with hypergraphs as a special data structure has been broadly studied in the machine learning community, e.g., semi-supervised learning with hypergraph regularization (Zhou, Huang, and Schölkopf 2006), modeling label correlations via hypernetworks in multi-label learning (Sun, Ji, and Ye 2008), and modeling communities to improve recommender systems (Bu et al. 2010). Zhou et al. (Zhou, Huang, and Schölkopf 2006) studied spectral clustering in hypergraphs. They generalized the normalized cut (Shi and Malik 2000) algorithm to hypergraph clustering and proposed a hypergraph Laplacian. They also proposed a semi-supervised hypergraph vertex classification algorithm leveraging hyperlink regularization. These research mainly aim to improve the learning performance on nodes by leveraging their hyperlink relations. However, none of them focuses on predicting the hyperlink relations. When dealing with hyperlink relations, existing research typically reduce hyperlinks to ordinary edges by clique expansion or star expansion (Agarwal, Branson, and Belongie 2006), which break the structure of a hyperlink as a whole.

We notice that hyperlink prediction is similar to the problem of selecting a good column subset (Boutsidis, Mahoney, and Drineas 2009). However, subset selection algorithms focus on selecting columns which best "capture " the candidate columns U, while hyperlink prediction requires the selected columns to best fit into the observed network S.

Experimental Results

In this section, we evaluate the effectiveness of the proposed Coordinated Matrix Minimization (CMM) algorithm on two novel tasks: predicting recipes of traditional Chinese food, and finding missing reactions of organisms' metabolic networks, both of which exemplify the application scenarios of hyperlink prediction. All the codes and data are available at https://github.com/muhanzhang/HyperLinkPrediction.

Predicting recipes

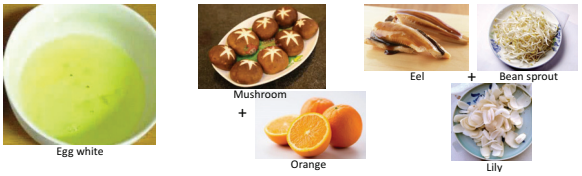
To visualize CMM's practical hyperlink prediction quality, we consider a recipe prediction problem: suppose we have a repository of cooking materials, which combinations of materials can produce delicious dishes? Given a hypernetwork of recipes where each node is a material and each hyperlink is a combination of materials that constitute a dish, we aim to predict new dishes based on the existing dishes.

Traditional Chinese dishes have a long history. Thousands of different dishes have been developed with various colors, aromas, and tastes, including the popular ones such as "Peking Duck", "Spring Rolls", "Kung Pao Chicken", and "Ma Po Tofu". There are different styles of Chinese cuisine based on regions. In this paper, we study the Sichuan cuisine and the Cantonese cuisine. We downloaded 882 most popular Sichuan recipes and Cantonese recipes from meishij.net, which is a professional platform to find Chinese recipes. After removing duplicated recipes, we have 725 Sichuan recipes (with 439 different materials) and 835 Cantonese recipes (with 500 different materials). For each cuisine, we delete 400 recipes and keep the remaining ones as the observed hyperlinks. We further randomly generate 1000 fake recipes according to the material distribution of the existing recipes, and combine them with the 400 real recipes to construct the set of candidate hyperlinks.

For evaluation, we rank all candidate hyperlinks with their scores Λ, select the top 400 hyperlinks as predictions, and report how many of them are real missing recipes. We also report the AUC (area under the ROC curve) scores measuring how likely a random real recipe is ranked higher than a random fake one. The number of latent factors k in CMM is set to 30. For Sichuan cuisine, our method can successfully predict 170 real recipes in the top 400 predictions, with an AUC score 0.6368. For Cantonese cuisine, our method successfully predicts 178 real recipes, with an AUC score 0.6608. For comparison, we test an information retrieval method, Bayesian Set (Ghahramani and Heller 2006), which is explained in the next experiment. Bayesian Set only predicts 123 and 98 recipes, with AUC scores 0.5014 and 0.4463 respectively. Our method significantly outperforms Bayesian Set in both number of correct predictions and AUC.

We visualize the top 1-material, 2-material, and 3-material predictions of both CMM and Bayesian Set for the Cantonese recipe prediction task in Figure 2. Our CMM predicts "Egg drop soup", "Double skin milk with cherry", and "Prawn and bamboo shoots egg soup", which are all real recipes. In comparison, Bayesian Set returns created recipes "Egg white", "Mushroom + Orange", and "Eel + Bean sprout + Lily", which are all strange combinations in the sense of Chinese cuisine. The failure of Bayesian Set for hyperlink prediction is because it treats hyperlinks as binary vectors and retrieves candidate hyperlinks whose binary vectors are most similar to those of the existing hyperlinks. This similarity is measured element-wise by assuming independent Bernoulli distributions of materials, which fails to capture the correlations among materials. In contrast, CMM does not aim to find hyperlinks similar to existing ones, but predict hyperlinks that are most suitable to fit into the observed hypernetwork. By modeling hyperlinks in the adjacency space, CMM also naturally considers the correlations between materials.

(a) Top predictions by CMM.



(b) Top predictions by Bayesian Set.

Figure 2: Top 1-material, 2-material, and 3-material predictions of Cantonese cuisine.

Figure 3: Created recipes by CMM.

We further examine the false positive predictions of CMM. To our surprise, many of them are indeed meaningful dishes. For example, CMM predicts "Flour + Dutch milk" (which can be used to make "Milk-flavored golden rolls"), "Coconut milk + Egg" (which can be used to make "Coconut milk egg custard"), and "Egg + Minced meat" (which can be used to make "Scrambled eggs with meat") etc. We illustrate these created recipes in Figure 3. Although these dishes do not exist in the downloaded recipes, our method successfully predicts them. This shows that our method is able to create meaningful recipes as well.

Dataset	Species	Vertices	Hyperlinks
(a) iJO1366	E. coli	1805	2583
(b) iAF1260b	E. coli	1668	2388
(c) iAF692	M. barkeri	628	690
(d) iHN637	Cl. Ljungdahlii	698	785
(e) iIT341	H. pylori	485	554
(f) iAB_RBC_283	H. sapiens	342	469

Table 1: Statistics of the six metabolic networks.

Predicting metabolic reactions

Reconstructed metabolic networks are important tools for understanding the metabolic basis of human diseases, increasing the yield of biologically engineered systems, and discovering novel drug targets (Bordbar et al. 2014). Semi-automated procedures have been recently developed to reconstruct metabolic networks from annotated genome sequences (Thiele and Palsson 2010). However, these networks are often incomplete – some vital reactions can be missing from them, which can severely impair their utility (Kumar, Dasika, and Maranas 2007). Thus, it is critical to develop computational methods for completing metabolic networks. Our task here is to find these missing reactions, which can be elegantly modeled as a hyperlink prediction problem, where each reaction is regarded as a hyperlink connecting its participating metabolites. Note that this systems biology problem is never studied using a statistical approach before. Previous approaches are based on gap-filling algorithms (Thiele, Vlassis, and Fleming 2014) designed to add reactions to an almost complete network to fill its functional gaps, which lack the ability to recover a very incomplete network in its initial reconstruction phase.

Datasets To evaluate the performance of CMM on finding missing metabolic reactions, we conduct experiments on six metabolic networks from five species: E. coli, M. barkeri, Cl. ljungdahlii, H. pylori and H. sapiens. The statistics of each dataset are shown in Table 1. We downloaded all 11893 reactions from BIGG (http://bigg.ucsd.edu) to build a candidate reaction pool. These reactions are collected from 79 metabolic networks of various organisms. We filter out the candidate reactions which contain exotic metabolites or already exist in the network.

For each metabolic network, we randomly delete some reactions as missing hyperlinks, and keep the remaining ones as the observed data. The numbers of deleted reactions range from 25 to 200 or from 50 to 400 according to network size.

We evaluate the reaction prediction performance using AUC as one measure. We also use a second measure: when N reactions are missing, we look at how many of the top-N predictions are true positive. We call the second measure "Number of recovered reactions". Compared to AUC, this measure only focuses on top predictions and better reflects practical reaction prediction performance.

Baselines and experimental setting Although hyperlink prediction is a fairly new problem, we come up with a wide range of promising baseline methods explained as follows.

BS (Bayesian Set) is an IR algorithm in the Bayesian framework. It takes a query consisting of a small set of items and returns additional items that belong in this set (Ghahramani and Heller 2006). Given a query $S = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\}$, BS computes score$(\mathbf{u}) = \frac{p(\mathbf{u}|S)}{p(\mathbf{u})}$ for all $\mathbf{u} \in U$ and retrieves hyperlinks with the highest scores. For $\mathbf{u}$ and $\mathbf{s}$, we assume each of their elements has an independent Bernoulli distribution with a common Beta prior distribution and use the default hyperparameters.

SHC (Spectral Hypergraph Clustering) is a state-of-the-art hypergraph learning algorithm (Zhou, Huang, and Schölkopf 2006). SHC outputs classification scores by $f = (I - \xi\Theta)^{-1}y$. The hyperparameter ξ is determined by searching over the grid $\{0.01, 0.1, 0.5, 0.99, 1\}$ using cross validation. SHC is originally designed to classify hypergraph vertices leveraging their hyperlink relations. Here we transpose the incidence matrices to change each vertex into a hyperlink and each hyperlink into a vertex, making SHC feasible for hyperlink prediction.

HPLSF is a hyperlink prediction method using supervised learning (Xu, Rockmore, and Kleinbaum 2013). It calculates an entropy score along each latent feature dimension in order to get a fixed-length feature input. We train a logistic regression model on these entropy features in order to output prediction scores.

FM (Factorization Machine) (Rendle 2012) is a flexible factorization model. We use the classification function of FM, where columns of the observed incidence matrix are used as input features to the model.

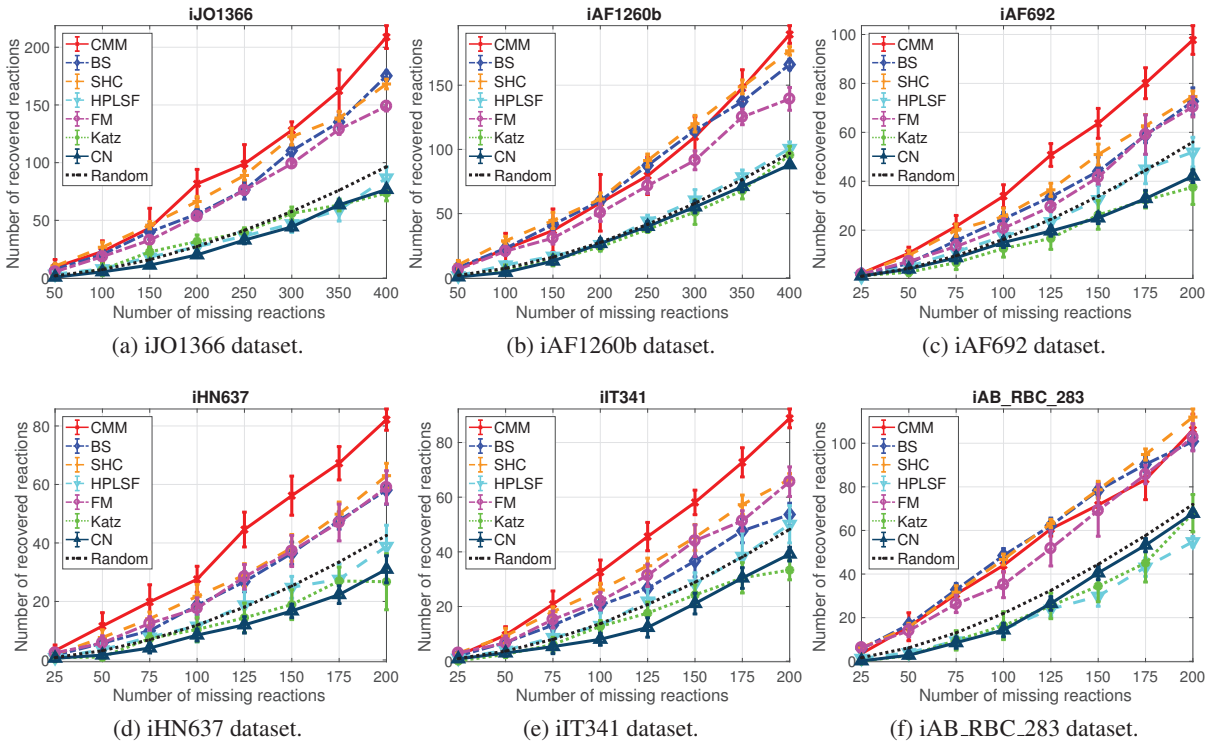
Katz generalizes the traditional pairwise Katz index (Katz 1953) to hyperlinks. Concretely, a hyperlink containing m vertices will have $m(m-1)/2$ pairwise Katz indices. We calculate their average as the hyperlink Katz index. The damping factor β is determined by searching over $\{0.001, 0.005, 0.01, 0.1, 0.5\}$ using cross validation.

CN generalizes the traditional pairwise common neighbors (Liben-Nowell and Kleinberg 2007) to hyperlinks, which follows a similar calculation to Katz.

Random: a theoretical baseline for comparing algorithms' performance against random. It is equal to assigning random scores between [0,1] to all candidate hyperlinks.

We implement the proposed CMM in MATLAB. We searched the latent feature number k in $\{10, 20, 30\}$ for small datasets by cross validation. For datasets (a) and (b), k was set to the default 30. The maximum iteration number was set to 100. The convergence threshold was set to 1.0E-4. All experiments were done on a 12-core Intel Xeon Linux server. All experiments were repeated 12 times and the average results and standard deviations are presented.

Results We first show the number of recovered reactions in Figure 4. CMM generally achieves the best performance. We observe that CMM recovers a significantly larger number of reactions than other baselines in datasets (a), (c), (d) and (e), and achieves highly competitive performance with the best baselines in datasets (b) and (f). The large proportion of true positive predictions can greatly reduce the network reconstruction effort by providing biologists the most likely reactions for later individual checking. We attribute the superior performance of CMM to the following reasons:



(a) iJO1366 dataset. (b) iAF1260b dataset. (c) iAF692 dataset.

(d) iHN637 dataset. (e) iIT341 dataset. (f) iAB_RBC_283 dataset.

Figure 4: Number of recovered reactions under different numbers of missing reactions.

Dataset	CMM	BS	SHC	HPLSF	FM	Katz	CN
(a)	0.7092±0.0180	0.6817±0.0082	**0.7105±0.0042**	0.4834±0.0335	0.6309±0.0228	0.5438±0.0178	0.4371±0.0105
(b)	0.7021±0.0034	0.6698±0.0131	**0.7150±0.0050**	0.5418±0.0088	0.6149±0.0142	0.4990±0.0320	0.4679±0.0028
(c)	**0.7035±0.0260**	0.5056±0.0295	0.6165±0.0178	0.4719±0.0450	0.5465±0.0212	0.4486±0.0177	0.4300±0.0213
(d)	**0.7050±0.0328**	0.5258±0.0265	0.6170±0.0138	0.4711±0.0500	0.5786±0.0198	0.4845±0.0214	0.4240±0.0214
(e)	**0.6794±0.0148**	0.5114±0.0231	0.5978±0.0117	0.5212±0.0471	0.5692±0.0180	0.4254±0.0362	0.4399±0.0100
(f)	**0.7098±0.0482**	0.6087±0.0144	0.6963±0.0122	0.4351±0.0126	0.6620±0.0275	0.5529±0.0195	0.3881±0.0116

Table 2: AUC results.

1) CMM makes inference in the adjacency space, which avoids directly performing inference in the incidence space that has size $\mathcal{O}(2^n)$. This transforms an $\mathcal{O}(2^n)$ problem into an $\mathcal{O}(n^2)$ problem, which greatly reduces the problem size and also addresses the variable cardinality problem; 2) CMM jointly optimizes the indicator matrix Λ and the latent factor matrix W. It simultaneously finds a subset of candidate hyperlinks that fit the network best as well as a latent factor matrix that explains the network best. The joint optimization procedure is derived from an EM optimization framework.

Now we analyze why the baselines do not perform well for hyperlink prediction. Firstly, as we have explained, BS as an information retrieval algorithm is not suitable for hyperlink prediction, as it retrieves similar items instead of unseen hyperlinks. For example, when encountering a candidate hyperlink already in the query, BS will give it a high score for being similar to the query, while CMM knows there is already a same one in the network and is more likely to reject it for other unseen hyperlinks which can complete the network. SHC has a reasonable performance on (a), (b), and (f), but is not comparable to CMM on (c), (d), and (e), since SHC is originally a node classification algorithm leveraging hyperlink relations, but not a hyperlink prediction algorithm.

HPLSF and FM are two classifier-based baselines which directly infer hyperlinks in the incidence space. Not surprisingly, they have much worse performance than CMM which predicts hyperlinks in the adjacency space. This also implies that hyperlink prediction is not suitable to be modeled as a classification problem – the special problem structure and the sparsity in hyperlinks require novel modeling schemes.

Katz and CN are two naive generalizations of traditional link prediction heuristics. Their poor performance (often worse than random guess) suggests that hyperlink prediction is not a simple generalization of link prediction, but is a **significantly harder** new problem. We may need to design new suitable heuristics for hyperlink prediction, or try to learn hyperlink prediction heuristics automatically from hypernetworks as suggested in (Zhang and Chen 2017).

We further report the average final AUC performance (the results under 400 or 200 missing reactions) in Table 2. The AUC results are generally consistent with the numbers of recovered reactions.

Conclusions

In this paper, we have considered the novel problem of predicting hyperlinks from a hypernetwork. Hyperlink prediction is an interesting and challenging problem. We have proposed a novel algorithm, Coordinated Matrix Minimization (CMM), leveraging an EM optimization framework. CMM first projects all hyperlinks into the adjacency space, and then simultaneously finds the candidate hyperlinks that best fit the network as well as the latent features that best explain the network. We have conducted comprehensive evaluation by comparing CMM with a wide range of baselines on two novel tasks. Experimental results demonstrated that our CMM algorithm is better than all the baseline methods.

Acknowledgments

The work is supported in part by the DBI-1356669, SCH-1343896, III-1526012, and SCH-1622678 grants from the National Science Foundation and grant 1R21HS024581 from the National Institute of Health.

References

Agarwal, S.; Branson, K.; and Belongie, S. 2006. Higher order learning with graphs. In *Proceedings of the 23rd international conference on Machine learning*, 17–24. ACM.

Bautu, E.; Kim, S.; Bautu, A.; Luchian, H.; and Zhang, B.-T. 2009. Evolving hypernetwork models of binary time series for forecasting price movements on stock markets. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, 166–173. IEEE.

Bordbar, A.; Monk, J. M.; King, Z. A.; and Palsson, B. O. 2014. Constraint-based models predict metabolic and associated cellular functions. *Nature Reviews Genetics* 15(2):107–120.

Boutsidis, C.; Mahoney, M. W.; and Drineas, P. 2009. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, 968–977. Society for Industrial and Applied Mathematics.

Bu, J.; Tan, S.; Chen, C.; Wang, C.; Wu, H.; Zhang, L.; and He, X. 2010. Music recommendation by unified hypergraph: combining social media information and music content. In *Proceedings of the 18th ACM international conference on Multimedia*, 391–400. ACM.

Chen, Z.; Chen, M.; Weinberger, K. Q.; and Zhang, W. 2015. Marginalized denoising for link prediction and multi-label learning. In *AAAI*, 1707–1713. Citeseer.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)* 1–38.

Ghahramani, Z., and Heller, K. A. 2006. Bayesian sets. In *Advances in neural information processing systems*, 435–442.

Karypis, G.; Aggarwal, R.; Kumar, V.; and Shekhar, S. 1999. Multilevel hypergraph partitioning: applications in vlsi domain. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 7(1):69–79.

Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18(1):39–43.

Kuang, D.; Ding, C.; and Park, H. 2012. Symmetric nonnegative matrix factorization for graph clustering. In *Proceedings of the 2012 SIAM international conference on data mining*, 106–117. SIAM.

Kumar, V. S.; Dasika, M. S.; and Maranas, C. D. 2007. Optimization based automated curation of metabolic reconstructions. *BMC bioinformatics* 8(1):1.

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58(7):1019–1031.

Lü, L., and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390(6):1150–1170.

Menon, A. K., and Elkan, C. 2011. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 437–452.

Miller, K.; Jordan, M. I.; and Griffiths, T. L. 2009. Nonparametric latent feature models for link prediction. In *Advances in neural information processing systems*, 1276–1284.

Oyetunde, T.; Zhang, M.; Chen, Y.; Tang, Y.; and Lo, C. 2016. Boostgapfill: improving the fidelity of metabolic network reconstructions through integrated constraint and pattern-based methods. *Bioinformatics* 33(4):608–611.

Rendle, S. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3(3):57.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22(8):888–905.

Song, D.; Meyer, D. A.; and Tao, D. 2015. Top-k link recommendation in social networks. In *Data Mining (ICDM), 2015 IEEE International Conference on*, 389–398. IEEE.

Sun, L.; Ji, S.; and Ye, J. 2008. Hypergraph spectral learning for multi-label classification. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 668–676. ACM.

Thiele, I., and Palsson, B. Ø. 2010. A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nature protocols* 5(1):93–121.

Thiele, I.; Vlassis, N.; and Fleming, R. M. 2014. fastgapfill: efficient gap filling in metabolic networks. *Bioinformatics* 30(17):2529–2531.

Wu, L.; Ge, Y.; Liu, Q.; Chen, E.; Long, B.; and Huang, Z. 2016. Modeling users preferences and social links in social networking services: A joint-evolving perspective. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Xu, Y.; Rockmore, D.; and Kleinbaum, A. M. 2013. Hyperlink prediction in hypernetworks using latent social features. In *Discovery Science*, 324–339. Springer.

Zhang, M., and Chen, Y. 2017. Weisfeiler-lehman neural machine for link prediction. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 575–583. ACM.

Zhou, D.; Huang, J.; and Schölkopf, B. 2006. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in neural information processing systems*, 1601–1608.