Monomer: Non-Metric Mixtures-of-Embeddings for Learning Visual Compatibility Across Categories

Ruining He  
UC San Diego  
r4he@ucsd.edu

Charles Packer  
UC San Diego  
cpacker@ucsd.edu

Julian McAuley  
UC San Diego  
jmcauley@ucsd.edu

ABSTRACT

Identifying relationships between items is a key task of an online recommender system, in order to help users discover items that are functionally complementary or visually compatible. In domains like clothing recommendation, this task is particularly challenging since a successful system should be capable of handling a large corpus of items, a huge amount of relationships among them, as well as the high-dimensional and semantically complicated features involved. Furthermore, the human notion of “compatibility” that we need to capture goes beyond mere similarity: For two items to be compatible—whether jeans and a t-shirt, or a laptop and a charger—they should be similar in some ways, but systematically different in others.

In this paper we develop a method, Monomer, to uncover complicated and heterogeneous relationships between items. Recently, scalable methods have been developed that address this task by learning embeddings of the visual and textual characteristics of the products involved, but which ultimately depend on a nearest-neighbor assumption between the learned embeddings. Here we show that richer notions of compatibility can be learned, principally by relaxing the metricity assumption inherent in previous work, so as to uncover ways in which related items should be systematically similar, and systematically different. Quantitatively, we show that our system achieves state-of-the-art performance on large-scale compatibility prediction tasks, especially in cases where there is substantial heterogeneity between related items.

Keywords

Recommender Systems, Visual Compatibility, Metric Learning

1. INTRODUCTION

Identifying and understanding relationships between items is a key component of any modern recommender system. Knowing which items are ‘similar,’ or which otherwise may be substitutable or complementary, is key to building systems that can understand a user’s current context, recommend alternative items from the same style [9], or generate bundles of items that are compatible [15, 20, 31].

Typically, identifying these relationships means defining (or otherwise learning from training data) an appropriate distance or similarity measure between items. This is appropriate when the goal is to learn some notion of ‘equivalence’ between items, e.g. in order to recommend an item that may be a natural alternative to the one currently being considered. However, identifying such a similarity measure may be insufficient when there is substantial heterogeneity between the items being considered. For example, the characteristics that make clothing items, electronic components, or even romantic partners compatible exhibit substantial heterogeneity: for a pair of such items to be compatible they should be systematically similar in some ways, but systematically different in others.

Recently, a line of work has aimed to model such heterogeneous relationships, e.g. to model co-purchasing behavior between products based on their visual appearance or textual descriptions [19, 20, 28]. In spite of the substantial heterogeneity in the data used for training (a large dataset of co-purchase ‘dyads’ from Amazon) and the complexity of the models used, these works ultimately follow an established metric-learning paradigm: (1) Collect a large dataset of related (and unrelated) items; (2) Propose a parameterized similarity function; and (3) Train the parameterized function such that related items are more similar than non-related items. Such metric-learning approaches can be incredibly flexible and powerful, and have been used to identify similarities between items ranging from music [24] to members of the same tribe [6]. Such methods work to some extent even in the presence of heterogeneity, since they learn to ‘ignore’ dimensions under which similarity should not be preserved. But we argue that ignoring such dimensions discards valuable information that ought to be used for prediction and recommendation.

In this paper, we propose new models and algorithms to identify relationships between items in product recommendation settings. In particular, we relax the metricity assumption present in recent work, by proposing more flexible notions of ‘relatedness’ while maintaining the same levels of speed and scalability. Specifically, we hope to overcome the following difficulties faced by previous approaches:

- Previous approaches rely on ‘nearest-neighbor’ assumptions to generate recommendations (albeit potentially via a complex embedding). This means that cross-category recommendations can only be made by exploiting an explicit category tree (e.g. ‘find the shoes nearest to these jeans’). Not only does this require an explicit category tree to be available, but it is also subject to whatever noise or deficiencies are present in the category data. Our method can make cross-category recommendations without any dependence on the presence (or quality) of explicit category information.
2. RELATED WORK

The most closely related branches of work to ours are (1) Those that deal with the item-to-item recommendation problem, e.g., systems that generate recommendations by modeling relationships between items; and (2) Works that deal with metric (or relationship) learning in general, whether or not for recommendation.

Item-to-Item Recommendation. Identifying relationships among items is a fundamental part of many real-world recommender systems, e.g., to generate recommendations of the form ‘people who viewed x also viewed y’ on Amazon. Such methods may be based on collaborative filtering, i.e., by counting the overlap between users who have clicked on / bought both items, as in Amazon’s own solution [16]. Latent-factor approaches aim to model user-item relationships in terms of low-dimensional factors, such that ‘similar’ items are those with close embeddings. See (e.g.) [11, 5] for surveys, and [18, 9], and [31] for specific systems that make use of Amazon, Etsy, and Ebay data (for example).

Of more interest to us are systems that predict item-to-item relationships based on the content (e.g., images/text/metadata) of the items themselves. Various systems have been proposed to address specific settings, e.g., to identify relationships between members of ‘urban tribes’ [22], tweets [25], text [2, 4], or music [24]. Several methods have also been used to model visual data [3, 13, 23, 20], though typically in settings (e.g., similar image retrieval) where the metric assumption is well-founded.

Our work follows recent examples that aim to model co-purchase and co-browsing relationships, using a recently introduced dataset from Amazon [19, 20, 28]. While we extend (and compare quantitatively against) such work, our main contribution here is that we substantially relax the model assumptions to allow for more complex relationships than mere similarity between items.

Also of interest are a few works that model clothing data, particularly in recommendation settings, e.g., a few recent works that aim to capture some notion of ‘style’ include [7, 11, 4, 17, 30]. However the specific tasks considered there are quite different from the item-to-item recommendation task in which we are interested.

Metric (and non-metric) Learning. Outside of the recommendation scenarios considered here, learning the features that describe relationships between objects is a vast topic. Typically, one is given some collection of putative relationships between items (i.e., a training set), and the goal is then to identify a (parameterized) function that can be tuned to fit these relationships, i.e., to assign observed relationships a higher likelihood or score than non-relationships. State-of-the-art methods identify hidden variables or factors that describe relationships among items [2], by factorizing the matrix of links between items [21]. Again, the main contributions we hope to make over such approaches are (1) to relax the assumption of metricity, and (2) to allow for multiple notions of ‘relatedness’ to compete and interact. While a few approaches have recently been proposed to learn non-metric relationships (e.g., [3]), we are unaware of any that allow for the scale of the data (thousands of features, millions of items and relationships) that we consider.

3. THE MODEL

We begin by formulating our task, before briefly introducing the preliminaries of our model. Then we provide details as to how we build and train a scalable model to address the challenges identified.

3.1 Formulation

We are given a dataset \( D \) comprising a large corpus of objects (or ‘items’) and the pairwise relationships \( R \) between items from
denote the matrix that parameterizes the Mahalanobis Distance, transforms as the basic building blocks of our method. The relationships between pairs of items. To this end, following informations that are capable of relating feature dimensions to explain 'styles' upon the raw visual features, we need expressive transformations.

In order to model subtle notions like styles, we apply a deep convolutional network pre-trained on 1.2 million ImageNet (ILSVRC2010) images. In particular, we used the Caffe reference model [12], which has 5 convolutional layers and 3 following fully-connected layers, to extract $F = 4096$ dimensional visual features from the second fully-connected layer (i.e., FC7).

Note however that our proposed method is agnostic to the type of features used, and as we show later can handle other types of features (e.g. text) in order to address more general settings.

**Mahalanobis Transform.** In order to model subtle notions like 'styles' upon the raw visual features, we need expressive transformations that are capable of relating feature dimensions to explain the relationships between pairs of items. To this end, following the approach from [20], we adopt the idea of using Mahalanobis transforms as the basic building blocks of our method.

A **Mahalanobis Distance** is able to measure the distance (or 'dissimilarity') of items within the feature space according to the knowledge of how different feature dimensions relate to each other. Let $M$ denote the matrix that parameterizes the Mahalanobis Distance, then the distance between an item pair $(x, y)$ is defined by

$$d_M(x, y) = (f_x - f_y)^T M (f_x - f_y),$$

(1)

where $f_x$ and $f_y$ are the (visual) features of item $x$ and $y$ respectively.

**Mixtures-of-Experts.** Mixtures of experts (MoEs) are a classical machine learning method to aggregate the predictions of a set of 'weak' learners, known as experts [10]. What is particularly elegant about this approach is that it allows each learner to 'focus' on classifying instances about which it is expert, without being penalized for making misclassifications elsewhere.

For regression tasks such as the one we consider in our model, each learner (denoted by $l$) outputs a prediction value $Pred_l(X)$ for the given input $X$. These predictions are then aggregated to generate the final prediction by associating weighted 'confidence' scores with each learner. Here we are interested in probabilistically modeling such confidences to be proportional to the expertise of the learners. Formally, this is expressed as

$$Pred(X) = \sum_l P(l|X)Pred_l(X),$$

(2)

In our model, each 'expert' shall correspond to a single notion of 'relatedness' between items. Thus, for a given pair of items that are potentially related, we can determine (a) which notions of relatedness are relevant for these items ($P(l|X)$); and (b) whether or not they are related according to that notion ($Pred_l(X)$). These two functions are learned jointly, such that the model automatically uncovers multiple notions of 'relatedness' simultaneously.

### 3.3 Model Specifics

In this subsection, we describe the specifics of our model. First we describe how Mahalanobis transforms have previously been applied to this task, and can be adopted by us as a building block. Then we discuss our proposed non-metric method (built upon the MoE framework) to compute the relatedness between item pairs.

#### 3.3.1 Low-rank Mahalanobis Metric

Considering the high dimensionality of the visual features we are modeling (feature dimension $F = 4096$ in our case), learning a full rank positive semi-definite matrix $M$ as in Eq. (1) is neither computationally tractable for existing solvers nor practical given the size of the dataset.

Recently it was shown in [20] that a low-rank approximation of a Mahalanobis matrix works very well on visual datasets for the tasks considered in this paper. Specifically, the $F \times F$ Mahalanobis matrix is approximated by $M \approx EE^T$, where $E$ is an $F \times K$ matrix and $K \ll F$. Then the distance between a pair $(x, y)$ is calculated by

$$d_E(x, y) = (f_x - f_y)^T EE^T (f_x - f_y) = ||E^T f_x - E^T f_y||_2^2.$$

(3)

This can be viewed as **embedding** the high-dimensional feature space ($F$-d) into a much lower-dimensional one ($K$-d) within which the Euclidean distance is measured. Note that the low rank property reduces the number of model parameters and increases the training efficiency significantly.

#### 3.3.2 Multiple, Non-Metric Embeddings

There are two key limitations from using a low-rank Mahalanobis embedding approach like the one above. First, it allows us to learn...
only a single set of dimensions (or the ‘statistically dominant reason’) that determines whether two given items are related or not. However, there might be multiple reasons relevant to the link discrimination in question. For example, a shirt and a pair of pants might go well together due to complementary colors, compatible textures, or simply some common characteristics they share (such as they both have pockets/buttons, etc.). This drives us to use a group of embeddings, parameterized by \( N \) matrices \( \mathbf{E}_1, \ldots, \mathbf{E}_N \) each with dimensionality \( F \times K \) for the prediction task, with each capturing a different set of factors or ‘reasons’ that items may be related.

Another limitation of the single Mahalanobis embedding method, or more generally any metric-based method, is that it assumes that the closest neighbor of a given item is always itself, which is inappropriate for our learning task of placing many different categories of item close to the target. To overcome this shortcoming, we propose to use a ‘support’ embedding (denoted by \( \mathbf{E}_0 \), again, with dimensionality \( F \times K \)) to learn the feature mappings in a non-metric manner.

In our model, \( \mathbf{E}_0 \) projects item \( x \) to a reference point \( \mathbf{E}_0^T f_x \) in the corresponding space, referred to as the ‘anchor’ space as it will be used as the basis for further comparisons. Next, embeddings \( \mathbf{E}_k \) (for \( k = 1, 2, \ldots, N \)) map the potential match \( y \) and correspond to a particular notion of relatedness, such that \( \mathbf{E}_k^T f_x \) will be close to \( \mathbf{E}_k^T f_y \) (for some \( k \)) if \( x \) and \( y \) are related. That is, the predicted distance \( d_k(x, y) \) by the \( k \)-th learner is

\[
d_k(x, y) = \left\| \mathbf{E}_k^T f_x - \mathbf{E}_k^T f_y \right\|^2.
\]

For clarity, we call the \( N \) spaces defined by \( \mathbf{E}_k \) \((k > 0)\) ‘pseudo’ spaces as all distance calculations are still performed within one actual space, i.e., the anchor space.

The above definition supports learning directed relationships as the model is not required to be symmetric; but, it is flexible enough learn symmetric (or even metric) embeddings if such structures are exhibited by the data.

### 3.3.3 Probabilistic Mixtures of Embeddings

Now we introduce how we aggregate the predictions from different embeddings. Given an item pair \( (x, y) \), we build our model upon the MoE framework to learn a probabilistic gating function to ‘switch’ among different embeddings. Considering our asymmetric setting where query item \( x \) in the pair is used as the reference point, we model the probability that the \( k \)-th embedding is used for the given an pair \((x, y)\) with a softmax formulation:

\[
P(k|\mathbf{x},\mathbf{y}) = \frac{\exp(U_k^T f_x)}{\sum_c \exp(U_c^T f_x)},
\]

where \( U \) is a newly-introduced \( F \times N \) parameter matrix with \( U_k \) being its \( k \)-th column. Briefly, the idea is to compute the probability distribution over the \( N \) learners given the characteristics of the ‘pivot’ item \( x \). Note that our formulation is efficient as it only introduces a small number of parameters given that \( N \) is usually a small number (e.g. on the order of 4 or 5 in our experiments).

![Figure 2: Illustration of the ‘distance’ \( d(x, y) \) between item \( x \) and \( y \) defined by Monomer. Each low-rank embedding (i.e., expert) learns a specific notion of relatedness, parameterized by an embedding matrix. The final output is a probabilistic sum of all learners’ predictions.](image)

Finally, our model calculates the ‘distance’ of an item pair \((x, y)\) by the probabilistic expectation:

\[
d(x, y) = \sum_{k=1}^{N} P(k|\mathbf{x},\mathbf{y}) \times d_k(x, y). \tag{6}
\]

Next, we need to randomly select a negative set of relationships \( \mathcal{R} \). To this end, we sample negative pairs in a way that (1) each item is associated with roughly same number of relationships and non-relationships, and (2) each negative pair consists of items from two categories.

Then we proceed by fitting the parameters by maximizing the log-likelihood of the training corpus:

\[
\hat{\Theta} = \arg \max_{\Theta} \mathcal{L}(\mathcal{R}, \mathcal{R}^\perp; \Theta) = \sum_{(x,y) \in \mathcal{R}} \log(P((x,y) \in \mathcal{R})) + \sum_{(x,y) \in \mathcal{R}^\perp} \log(1 - P((x,y) \in \mathcal{R}^\perp)) + \Omega(\Theta), \tag{7}
\]

Figure 2 summarizes the procedure introduced above. Note that our ‘distance’ definition is a non-metric method as it only preserves the non-negativity and is relaxing the symmetry, identity, and triangle inequality properties.

### 3.3.4 Log-likelihood Maximization

With the ‘distance’ function defined above, we model the probability that a pair is related by a shifted sigmoid function (in a way similar to [20]):

\[
P((x, y) \in \mathcal{R}) = \sigma_c(-d(x, y)) = \frac{1}{1 + \exp(d(x, y) - c)} \tag{8}
\]

where \( \Theta \) is the full parameter set \( \{\mathbf{E}_0, \mathbf{E}_1, \ldots, \mathbf{E}_N, \mathbf{U}, c\} \), and \( \Omega(\Theta) \) is an \( L_2 \)-regularizer to avoid overfitting. The total number of parameters is \( F \times (N \times K + K + N) + 1 \). Since \( N \) and \( K \) are small numbers, the log-likelihood as well as the derivatives can be computed efficiently.

### 3.4 Learning the Model

Our model is learned with L-BFGS [18], a quasi-Newton method for non-linear optimization of problems with a large number of variables. Note that our log-likelihood and the full derivative computations can be naïvely parallelized over all training pairs \((x, y) \in \mathcal{R} \cup \mathcal{R}^\perp\). See Appendix for the scalability analysis of our method.

\[ d_{ij}(x, y) \text{ is defined by Eq.\ 4 while the Probabilistic Gating function is defined by Eq\ 5}\]
4. EXPERIMENTS

In this section, we perform comprehensive experiments to evaluate our proposed method.

4.1 Dataset

To fully evaluate the performance of our method, we want to experiment on the largest dataset available to test the ability of Monomer to cope with real-world scenarios. To this end, we conduct all our experiments on the dataset from Amazon, recently introduced by [20]. We focus on five top-level categories under the category tree rooted with ‘Clothing & Jewelry’, i.e., Men’s, Women’s, Boys’, Girls’, and Baby’s Clothing & Accessories. Statistics are shown in Table 2.

For each of the above categories, we experiment with two important types of relationships: ‘users who bought x also bought y’, and ‘users who viewed x also viewed y’, denoted as ‘also_bought’ and ‘also_viewed’ respectively for brevity. These two types of relationships are a key source of data to learn from in order to recommend items of potential interests to customers. Ground-truth for these relationships are also introduced by the authors of [20], and are originally derived from co-purchase and co-browsing data from Amazon.

Recall that our objective is to learn heterogeneous relationships so as to support cross-category recommendation. Across the entire dataset, such relationships are noisy, sparse, and not always meaningful. To address issues of noise and sparsity to some extent, it’s sensible to focus on the relationships within the scope of a particular top-level category, e.g. Women’s Clothing, Men’s Clothing etc. We then consider relationships between ‘2nd-level’ categories, e.g. women’s shirts, women’s shoes etc.

In summary, our evaluation protocol is as the following:

1. A single experiment consists of a specific category (e.g. Men’s Clothing) and a graph type (e.g. ‘also_bought’).
2. For each experiment, the relationships (\(R\)) and a random sample of non-relationships (\(\bar{R}\)) are pairs of items connecting different subcategories of the category we are experimenting on. Note that \(|\bar{R}| = |\bar{R}|\) and they share the same distribution over the items.
3. For each experiment, we use an 80/10/10 random split of the dataset (\(R \cup \bar{R}\)) with the training set being at most two million pairs. Our goal is then to predict the relationships and non-relationships correctly, i.e., link prediction.
4. In all cases, the regularization hyperparameter is tuned with grid-search to perform the best on the validation set. Correspondingly, all results are reported on the test set.

For example, one experiment is to predict ‘also_bought’ relationships for Men’s Clothing. There are 56 subcategories under Men’s Clothing (see Table 2), so our goal is to distinguish edges from non-edges connecting items from among these subcategories.

All our experiments were performed on a single machine with 64GB main memory and 8 cores. Our largest experiment required around 40 hours of training time, though most were completed in a few hours.

4.2 Comparison Methods

We compare our method against the following baselines, following the experimental protocol introduced above.

**Weighted Nearest Neighbor (WNN):** This method uses a weighted Euclidean distance in the raw feature space to measure the similarity of two items: 
\[
d_{w}(x, y) = ||w \circ (f_x - f_y)||^2_2.
\]
Here \(\circ\) is the Hadamard product and \(w\) is a weighting vector that is learned from the data.

**Category Tree (CT):** This method computes a matrix of co-occurrences between subcategories from the training data. Then a pair \((x, y)\) is predicted to be positive if the subcategory of \(y\) is one of the top 50% most commonly connected subcategories to the subcategory of \(x\).

**Low-rank Mahalanobian Transform (LMT):** LMT is a state-of-the-art method for learning visual similarities among different items (possibly between categories) on large-scale datasets. LMT learns a single low-rank Mahalanobian embedding matrix to embed all items into a low-dimensional space. Then it predicts the links between a given pair based on the Euclidean distance within the embedded space (i.e., Eq. 4).

**Non-metric Mixtures-of-Embeddings (Monomer):** Our method. This method learns a mixture of low-rank transforms/embeddings to uncover groups of underlying reasons that explain the relationships between items. It measures the ‘distance’ (or dissimilarity) between items in an non-metric manner (i.e., Eq. 5).

Ultimately, our baselines are designed to demonstrate (a) that the raw feature space is not directly suitable for learning the notions of relationships (WNN); (b) the performance to be achieved by using the category metadata directly, but not using other features (CT); and (c) the improvement of our proposed model over the state-of-the-art method on our task (LMT).

To perform a fair comparison between LMT and Monomer, the following setting is used for all experiments in this paper:

1. It has been shown by [20] that LMT can achieve better accuracy when using a reasonably large number of embedding dimensions \(K\). Therefore in all cases we choose \(K\) large enough such that LMT obtains the best possible performance.
2. In all cases we try to compare LMT and Monomer under the same total number of model parameters. For example, if we set the number of dimensions \(K\) to 100 for LMT, then a fair setting for Monomer would be \(K = 20\) and \(N = 4\). This way both of them are using 100\(F\) embedding parameters. Recall that \(N\) is the number of embeddings (excluding the ‘support’ embedding).

4.3 Performance and Quantitative Analysis

Error rates on the test set for all experiments are reported in Table 3. For experiments on ‘also_bought’ relationships, LMT uses \(K = 100\) dimensions and Monomer uses \(K = 20\) and \(N = 4\). While for experiments on ‘also_viewed’ relationships, \(K\) is set to 50 for LMT and \(K = 10\) and \(N = 4\) for Monomer. Note that ‘also_viewed’ relationships are almost twice as sparse as ‘also_bought’ relationships (and thus a model with fewer parameters performed better at validation time), as shown in Table 3. We make a few observations to explain and understand our findings as follows:

| Dataset | #Subcategories | #Items | Relationship (#Edges) |
|---------|---------------|-------|----------------------|
| Men     | 56            | 306,215 | 1,075,547 | 635,610 |
| Women   | 116           | 659,566 | 1,923,952 | 1,691,121 |
| Boys    | 41            | 42,156  | 169,503  | 75,689  |
| Girls   | 44            | 56,593  | 191,964  | 97,881  |
| Baby    | 6             | 36,588  | 96,253   | 95,784  |
| Total   | 263           | 1,101,118 | 3,457,219 | 2,596,085 |
Figure 3: Visualization of Monomer trained on Women’s Clothing for ‘also_bought’ prediction. Each visual space is demonstrated by a grid view of 2-d t-SNE visualization (each cell randomly selects one image in overlapping cases). According to our ‘distance’ function (defined by Eq. 6), Monomer will recommend the nearest neighbors of the query within each visual space, based on the associated ‘reasons’ learned from the data. Note that in the example we are showing the full sets of nearest neighbors which might have been overlapped in the grid views.

Table 3: Test errors of the link prediction task using visual features (4096-d) for each edge type on clothing categories of the Amazon dataset. The best performing method in each case is boldfaced. Lower is better.

| Dataset | Graph       | (a) | (b) | (c) | (d) | impr. |
|---------|-------------|-----|-----|-----|-----|------|
|         | WNN         | CT  | LMT | Monomer | d vs. c |
| Men     | also_bought | 34.95% | 47.71% | 9.20% | 6.48% | 30%  |
|         | also_viewed | 30.50% | 49.73% | 11.52% | 7.87% | 32%  |
| Women   | also_bought | 31.16% | 46.02% | 8.80% | 5.71% | 35%  |
|         | also_viewed | 21.52% | 46.22% | 6.72% | 5.35% | 20%  |
| Boys    | also_bought | 31.10% | 47.63% | 8.33% | 5.78% | 31%  |
|         | also_viewed | 22.36% | 46.43% | 6.46% | 5.62% | 13%  |
| Girls   | also_bought | 37.26% | 48.01% | 12.48% | 7.94% | 36%  |
|         | also_viewed | 30.89% | 47.72% | 11.88% | 9.25% | 22%  |
| Baby    | also_bought | 37.26% | 48.01% | 12.48% | 7.94% | 36%  |
|         | also_viewed | 30.89% | 47.72% | 11.88% | 9.25% | 22%  |
| Avg.    | 27.92% | 47.64% | 9.00% | 6.79% | 22.9% |

1. WNN is particularly inaccurate for our task. We also observed relatively high training errors of this method for most experiments. This confirms our conjecture that raw similarity is inappropriate for our task, and that in order to learn the relationships across (sub)categories, some sort of expressive transforms are needed for manipulating the raw features.

2. The counting method (CT) performs considerably worse than other methods. This reveals that the predictive information used by the other models goes beyond the categories of the products, i.e., that the image-based models are learning relationships between finer-grained attributes.

3. Note that all models perform better at predicting ‘also_viewed’ than ‘also_bought’ relationships. This is reasonable since intuitively items that are “also viewed” indeed tend to share more common characteristics compared to the “also bought” scenario. The greater heterogeneity between training pairs in the latter task makes it comparatively harder to address.

4. Monomer outperforms LMT significantly for all experiments, especially for the harder task of predicting co-purchase dyads.

### 4.4 Visualization of the Embeddings

Next, we proceed by demonstrating the embeddings learned from our largest dataset, Women’s Clothing, by Monomer. We take the same model trained on co-purchase relationships from the previous subsection and visualize it in Figure 3. In this figure, we show each of the 5 visual spaces by a 2-d visualization with t-SNE. Images are a random sample of size 50,000 from the Women’s Clothing dataset and projected (using the learned embedding matrices) to each visual space to demonstrate the underlying structure.

As analyzed in Section 3, each embedding (i.e., learner) is capturing a specific notion of relatedness that explains the relationships of pairs of items in the corpus. In other words, it means that the nearest neighbors in each of the N ‘pseudo’ spaces should be related to the query according to the specific notion captured. Therefore those neighbors should be recommended as potential matches to the query item, as shown by the example in Figure 3. For the query image (a t-shirt) in this example, Monomer recommends bundles of similar t-shirts, pants, shoes, and accessories (watches etc.) that resemble the query in terms of patterns (e.g. space 1), colors (e.g. space 2), and more generally ‘styles’ (e.g. space 3 and 4). Such matching between a query image and nearby items in alternate spaces directly facilitates the task of recommending visually consistent outfits, where modeling and understanding the visual compatibility across categories is essential.
we mainly compare it to the state-of-the-art metric-based method, LMT. Both methods are able to learn relationships from the data, so one common setting is to retrieve ‘similar’ items (i.e., maximum probability of being related) to a given query.

First we train LMT and Monomer on Women’s Clothing to predict ‘also_bought’ relationships, under the same setting as in Table 3. This way the two models will learn their own own similarity (or distance) functions from the data. Next, from the Women’s Clothing we randomly select a few query items, for each of which LMT and Monomer will retrieve its highest-probability links according to their own similarity functions. Figure 5 demonstrates such queries and the retrieved connections (in all cases ranked in decreasing order in terms of the probability of the link) by the two models.

As shown by Figure 5 the metric-based method (LMT) tends to recommend items that are very similar to the query, even though for this task it is trained to predict complementary relationships (i.e., ‘also_bought’). Indeed it’s very difficult for a metric-based method to project items from different subcategories to be nearer than items from the same category; presumably such methods are limited by their underlying assumption that the most similar item to a given query is always itself. In [20] this was addressed to some extent by making explicit use of the category information at test time (e.g. ‘find the shirt closest to this pair of shoes’), though our model is able to make diverse sets of recommendations without such a dependence on explicit category information.

Recall that LMT learns an embedding within which the Euclidean distance is used to distinguish relationships from non-relationships. Visualizing such spaces can help understand the behavior of LMT. Again we uniformly sample 10,000 items from the dataset and use t-SNE [27] to visualize their positions in the embedded space. We are particularly interested in the distribution of different subcategories of items over the space. Therefore we assign a unique color to each subcategory in the dataset. Figure 6 shows results on two representative datasets, Men’s and Women’s Clothing.

From Figure 6 we find that subcategories of items tend to be projected to be ‘clusters’ in the embedded space. This can be a source of problems when performing recommendation tasks, especially when recommending related items from different subcategories:

1. From a recommender system’s perspective, there will be a ‘limited coverage’ issue because given a query LMT tends to recommend only those items on the boundaries of the clusters. There is no way that items located near the center of a cluster will ever be recommended.

2. Recommendations will suffer from ‘mislabeling’ issues. Note that our proposed non-metric method, Monomer, doesn’t suffer from either of above issues since it has already successfully ‘blended’ different subcategories, as shown by the nearest neighbors in Figure 5.

5. GENERALIZING TO BOW FEATURES

In previous sections, we have shown that Monomer not only performs very well on link prediction tasks but also that it recommends
highly diverse sets of items. However, above we only considered scenarios in which co-purchasing relationships can be predicted from visual features. Following this, a natural question would be “Is Monomer able to learn relationships from non-visual features and achieve similarly competitive performance?”

To answer the above question, we perform further experiments on Bag-of-Words (BoW) features extracted from the text of product reviews, which are also available in the Amazon dataset. In particular, for each category (e.g. Men’s Clothing) we use the following procedure to generate BoW features for all its items:

1. Remove stop-words and construct a dictionary. Our dictionary consists of 5000 nouns or adjectives or adjective-noun bigrams that appear most frequently in the review corpus being considered.

2. For each item $i$, a document $d_i$ is generated by bagging all the reviews it has received.

3. The 5000-d BoW feature vector $f_i$ of each item $i$ is computed by normalizing the raw word counts of document $d_i$ to sum to 1.

4. Items without any reviews attached are seen as invalid items and are dropped from the dataset. Note that this leaves us with fewer items (and thus fewer edges and non-edges) in the dataset.

In the following experiments, we use the same setting as in the previous visual feature experiments. Since our datasets are sparser than before, this time we fix $K = 50$ for LMT, and $K = 10$ and $N = 4$ for Monomer in all cases.

**Latent Dirichlet Allocation + WNN (LDA):** Here we add another baseline for further comparison. This method first obtains 100 topics with LDA with a vocabulary of size 5000, and then uses WNN to distinguish relationships from non-relationships within the 100-d topic space.

---

Table 4: Test errors of the link prediction task using textual features for each edge type on clothing categories of the Amazon dataset. For LMT, $K = 50$, while for Monomer, $K = 10$ and $N = 4$. The best performing method in each case is boldfaced. Lower is better.

| Dataset  | Graph | (a) WNN | (b) LDA | (c) LMT | (d) Monomer | impr. d vs. c |
|----------|-------|---------|---------|---------|-------------|--------------|
| Men      | also_bought | 45.85% | 40.22% | 21.79%  | 14.85%      | 32%          |
|          | also_viewed  | 42.26% | 34.65% | 20.75%  | 14.16%      | 32%          |
| Women    | also_bought | 44.63% | 36.54% | 21.50%  | 15.27%      | 29%          |
|          | also_viewed  | 42.07% | 30.87% | 18.75%  | 13.01%      | 31%          |
| Boys     | also_bought | 45.58% | 42.77% | 25.06%  | 14.67%      | 42%          |
|          | also_viewed  | 47.30% | 41.23% | 23.02%  | 17.12%      | 26%          |
| Girls    | also_bought | 46.05% | 45.31% | 27.32%  | 15.71%      | 43%          |
|          | also_viewed  | 47.64% | 46.72% | 25.04%  | 17.17%      | 31%          |
| Baby     | also_bought | 46.19% | 47.69% | 25.81%  | 14.75%      | 43%          |
|          | also_viewed  | 47.14% | 45.54% | 23.53%  | 19.93%      | 15%          |
| Avg.     |          | 45.47% | 41.15% | 23.26%  | 15.66%      | 32.4%        |

---

2 Statistics are suppressed due to limited space, though all data is available online.

3 We adopted the implementation in Gensim (default parameters kept): https://radimrehurek.com/gensim/
5.1 Results and Analysis

Table 4 summarizes the error rates on the test sets for all experiments. Again we observe that (1) generally speaking all models perform better for the ‘also_viewed’ prediction task, which is in line with our previous results; (2) basic methods like WNN and LMT are not particularly accurate for the task; and (3) Monomer outperforms LMT considerably especially on the harder tasks, which demonstrates its ability to handle textual features.

For the data we consider, BoW features are generally less discriminative than visual features, as we see from the comparison between Table 3 and Table 4. There are two main reasons for this: (1) It is because the datasets we are experimenting with so far concern clothing and accessories. Thus co-purchasing and co-browsing relationships can largely be explained through ‘visual’ reasons (e.g. sharing the same style), which means that visual features can be as or more expressive than BoW features for explaining such relationships. (2) Reviews are too short in certain cases (or there are too few reviews), which is especially true for cold-start items. When using visual features we suffer from no such issues, as even newly-introduced products always contain an image.

5.2 Non-clothing Categories

Next, we are concerned with whether Monomer can be applied to categories other than clothing. To answer this question, we performed more experiments on a variety of top-level Amazon categories (e.g. Electronics, Video Games, Home & Kitchen, etc.). Since visual factors play less of a role in these categories, we rely on BoW features extracted using the procedure described earlier. Statistics of these datasets are shown in the Appendix (Table 5).

Using the same evaluation protocol as for the clothing datasets, test errors are summarized in Table 5. Note that in all cases we set \( K = 100 \). For \( K = 20 \) and \( N = 4 \) for Monomer since we have comparatively more training instances than in previous cases. From the table we see that these results are consistent with those achieved on clothing data. There does exist one minor difference—the comparative hardness of ‘also_bought’ over ‘also_viewed’ prediction now seems to be dependent on the dataset in question, presumably due to different semantics of the two link types, or different patterns of customer behavior, among the different categories.

6. CONCLUSION

In this paper, we presented Monomer, a method to model heterogeneous relationships for item-to-item recommendation tasks. We noted that existing methods for item-to-item recommendation suffer from a few limitations when dealing with heterogeneous data, due mainly to their reliance on metricity or ‘nearest-neighbor’ type assumptions. To overcome these limitations, our method made use of ‘mixtures’ of embeddings, such that an item \( x \) is compatible with another item \( y \) if \( x \)’s embedding maps close to a different embedding of \( y \)’s features. This allows us to relax the identity and symmetry assumptions of existing metric-based methods, and to generate diverse and cross-category recommendations effectively. We showed quantitatively that Monomer is accurate at link prediction tasks using co-purchase and co-browsing dyads from Amazon, and qualitatively that it is able to generate diverse recommendations that are consistent with a particular visual style.

APPENDIX

A. SCALABILITY ANALYSIS

In Monomer, each embedding matrix has \( F \times K \) parameters, which means there are in total \( F \times K \times (N + 1) \) embedding parameters. As we have shown in our experiments, although Monomer is using multiple embedding matrices, it doesn’t need to use more embedding parameters to outperform the single-embedding baseline LMT. Therefore the following analysis focuses on comparing Monomer and LMT under the same total number of embedding parameters, in terms of the amount of multiplication operations involved.

For complete clarity, we denote the embedding dimension of LMT and Monomer by \( K' \) and \( K \) respectively \((F \times K' = F \times K \times (N + 1))\). For each training pair \((x, y)\), LMT takes \( O(F \times K') \) to compute the distance between them. Derivatives corresponding to this pair can also be calculated within \( O(F \times K') \) time. While for Monomer, it takes \( O(F \times K' \times K') \) to project \( x \) and \( y \) to the multiple spaces. Afterwards the ‘distance’ can be calculated in \( O(N \times K) + O(N \times F) \), where the former is used to compute \( N \) distance components and the latter is spent on the probabilistic weights (see Figure 2). To sum up, it takes \( O(F \times K' \times K') + O(N \times K) + O(N \times F) = O(F \times K' + K') \) for Monomer to finish ‘distance’ computation. Likewise, it’s easy to verify that the corresponding derivatives can also be computed in \( O(F \times K' \times K') \) time.

In summary, when using the same total number of embedding parameters, training Monomer and LMT will have the same worst-case time complexity. Note that visual features extracted from the deep convolutional neural network are sparse, which significantly reduces the actual training cost for both methods.

B. TABLES

Table 5 shows statistics of the top-level categories from the Amazon dataset used in Table 5.

References

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. TKDD, 2005.
Table 6: Statistics of a variety of top-level categories from the Amazon dataset.

| ID | Dataset            | #Subcategories | #Items | Relationship (#Edges) |
|----|--------------------|----------------|--------|-----------------------|
| Elect | Electronics       | 306            | 412,082 | 1,654,552 | 718,361 |
| Auto  | Automotive        | 178            | 312,642 | 959,353   | 1,298,774 |
| Games | Video Games       | 16             | 49,801  | 314,124   | 54,559  |
| Movies | Movies & TV       | 2              | 199,737 | 648,256   | 49,924  |
| Office | Office Products   | 245            | 127,054 | 448,720   | 370,630 |
| Home  | Home & Kitchen    | 81             | 393,781 | 560,574   | 960,925 |
| Phones | Cell Phones & Accessories | 28 | 317,965 | 867,418 | 225,785 |
|        |                    |                |        | **Total**             | 856     | 1,813,062 | 5,452,997 | 3,678,958 |

[2] R. Balasubramanyan and W. Cohen. Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In SDM, 2011.
[3] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. IEEE Trans. on PAMI, 2007.
[4] J. Chang and D. Blei. Relational topic models for document networks. In AISTats, 2009.
[5] S. Changpinyo, K. Liu, and F. Sha. Similarity component analysis. In NIPS, 2013.
[6] M. Der and L. Saul. Latent coincidence analysis: A hidden variable model for distance metric learning. In NIPS, 2012.
[7] W. Di, C. Wah, A. Bhardwaj, R. Piramuthu, and N. Sundaresan. Style finder: Fine-grained clothing style detection and retrieval. In CVPR Workshops, 2013.
[8] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. A. Efros. What makes paris look like paris? SIGGRAPH, 2012.
[9] D. J. Hu, R. Hall, and J. Attenberg. Style in the long tail: Discovering unique interests with latent variable models in large scale social e-commerce. In KDD, 2014.
[10] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. Neural computation, 3(1):79–97, 1991.
[11] Y. Jagadeesh, R. Piramuthu, A. Bhardwaj, W. Di, and N. Sundaresan. Large scale visual recommendations from street fashion images. arXiv:1401.1778, 2014.
[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In MM, 2014.
[13] X. Jin, J. Luo, J. Yu, G. Wang, D. Joshi, and J. Han. Reinforced similarity integration in image-rich information networks. IEEE Trans. on Knowledge and Data Engineering, 2013.
[14] Y. Kalantidis, L. Kennedy, and L.-J. Li. Getting the look: Clothing recognition and segmentation for automatic product suggestions in everyday photos. In ICMR, 2013.
[15] Y. Koren and R. Bell. Advances in collaborative filtering. In Recommender Systems Handbook. Springer, 2011.
[16] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, 2003.
[17] S. Liu, J. Feng, Z. Song, T. Zhang, H. Lu, C. Xu, and S. Yan. Hi, magic closet, tell me what to wear! In ACM Conference on Multimedia, 2012.
[18] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang. On centroidal voronoi tessellation—energy smoothness and fast computation. ACM Transactions on Graphics (ToG), 28(4):101, 2009.
[19] J. J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In KDD, 2015.
[20] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In SIGIR, 2015.
[21] A. K. Menon and C. Elkan. Link prediction via matrix factorization. In ECML, 2011.
[22] A. C. Murillo, I. S. Kwak, L. Bourdev, D. Kriegman, and S. Belongie. Urban tribes: Analyzing group photos from a social perspective. In CVPR Workshops, 2012.
[23] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. ACM Trans. of Graphics, 2011.
[24] M. Slaney, K. Weinberger, and W. White. Learning a metric for music similarity. In International Conference of Music Information Retrieval, 2008.
[25] D. Spina and J. Gonzalo. Learning similarity functions for topic detection in online reputation monitoring. In SIGIR, 2014.
[26] L. Torresani and K. chi-hoa Lee. Large margin component analysis. In NIPS, 2007.
[27] L. van der Maaten. Accelerating t-SNE using tree-based algorithm. JMLR, 2014.
[28] A. Veit, B. Kovacs, S. Bell, J. McAuley, K. Bala, and S. Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In ICCV, 2015.
[29] H. Xia, P. Wu, S. Hui, and R. Jin. Boosting multi-kernel locality-sensitive hashing for scalable image retrieval. In SIGIR, 2012.
[30] K. Yamaguchi, M. H. Kiapour, and T. L. Berg. Paper doll parsing: Retrieving similar styles to parse clothing items. In ICCV, 2013.
[31] J. Zheng, X. Wu, J. Niu, and A. Bolivar. Substitutes or complements: another step forward in recommendations. In EC, 2009.