

# Compatibility Family Learning for Item Recommendation and Generation

Yong-Siang Shih,<sup>1</sup> Kai-Yueh Chang,<sup>1</sup> Hsuan-Tien Lin,<sup>1,2</sup> Min Sun<sup>3</sup>

<sup>1</sup>Appier Inc., Taipei, Taiwan,

<sup>2</sup>National Taiwan University, Taipei, Taiwan,

<sup>3</sup>National Tsing Hua University, Hsinchu, Taiwan.

{yongsiang.shih,kychang}@appier.com, htlin@csie.ntu.edu.tw, sunmin@ee.nthu.edu.tw

## Abstract

Compatibility between items, such as clothes and shoes, is a major factor among customer's purchasing decisions. However, learning "compatibility" is challenging due to (1) broader notions of compatibility than those of similarity, (2) the asymmetric nature of compatibility, and (3) only a small set of compatible and incompatible items are observed. We propose an end-to-end trainable system to embed each item into a latent vector and project a query item into  $K$  compatible prototypes in the same space. These prototypes reflect the broad notions of compatibility. We refer to both the embedding and prototypes as "Compatibility Family". In our learned space, we introduce a novel Projected Compatibility Distance (PCD) function which is differentiable and ensures diversity by aiming for at least one prototype to be close to a compatible item, whereas none of the prototypes are close to an incompatible item. We evaluate our system on a toy dataset, two Amazon product datasets, and Polyvore outfit dataset. Our method consistently achieves state-of-the-art performance. Finally, we show that we can visualize the candidate compatible prototypes using a Metric-regularized Conditional Generative Adversarial Network (MrCGAN), where the input is a projected prototype and the output is a generated image of a compatible item. We ask human evaluators to judge the relative compatibility between our generated images and images generated by CGANs conditioned directly on query items. Our generated images are significantly preferred, with roughly twice the number of votes as others.

## 1 Introduction

Identifying compatible items is an important aspect in building recommendation systems. For instance, recommending matching shoes to a specific dress is important for fashion; recommending a wine to go with different dishes is important for restaurants. In addition, it is valuable to visualize what style is missing from the existing dataset so as to foresee potential matching items that could have been up-sold to the users. We believe that the generated compatible items could inspire fashion designers to create novel products and help our business clients to fulfill the needs of customers.

For items with a sufficient number of viewing or purchasing intents, it is possible to take the co-viewing (or co-purchasing) records as signals of compatibility, and simply



Figure 1: Notion of Compatibility (Left) vs. Similarity (Right). **Left:** The upper outer garment in the center (red circle) is the query item. The surrounding items are its compatible ones. The styles of both the compatible shoes and lower body garments are various. **Right:** The style of a similar item (bottom) is constrained.

use standard techniques for a recommendation system, such as collaborative filtering, to identify compatible items. In real world application, it is quite often encountered that there are insufficient records to make a decent compatible recommendation — it is then critical to fully exploit relevant contents associated with items, such as the images for dresses, or the wineries for wines.

Even leveraging such relevant information, recommending or generating compatible items is challenging due to three key reasons. First, the notion of compatibility typically goes across categories and is broader and more diverse than the notion of similarity, and it involves complex many-to-many relationships. As shown in Figure 1, compatible items are not necessarily similar and vice versa. Second, the compatibility relationship is inherently asymmetric for real world applications. For instance, students purchase elementary textbooks before buying advanced ones, house owners buy furniture only after their house purchases. Recommendation systems must take the asymmetry into consideration, as recommending car accessories to customers who bought cars is rational; recommending cars to those who bought car accessories would be improper. The two reasons above make many existing methods (McAuley et al. 2015; Veit et al. 2015; Lee, Seol, and Lee 2017) less fit for compatibility learning, as they aim to learn a symmetric metric so as to model the item-item relationship. Third, the currently available labeled data sets of compatible and incom-

patible items are insufficient to train a decent image generation model. Due to the asymmetric relationships, the generator could not simply learn to modify the input image as most CGANs do in the similarity learning setting.

However, humans have the capabilities to create compatible items by associating internal concepts. For instance, fashion designers utilize their internal concept of compatibility, e.g., style and material to design many compatible outfits. Inspired by this, we demonstrate extracting meaningful representation from the image contents for compatibility is an effective way of tackling such challenges.

We aim at recommending and generating compatible items through learning a “Compatibility Family”. The family for each item contains a representation vector as the embedding of the item, and multiple compatible prototype vectors in the same space. We refer to the latent space as the “Compatibility Space”. Firstly, we propose an end-to-end trainable system to learn the family for each item. The multiple prototypes in each family capture the diversity of compatibility, conquering the first challenge. Secondly, we introduce a novel Projected Compatibility Distance (PCD) function which is differentiable and ensures diversity by encouraging the following properties: (1) at least one prototype is close to a compatible item, (2) none of the prototypes is close to an incompatible item. The function captures the notion of asymmetry for compatibility, tackling the second challenge. While our paper focuses mainly on image content, this framework can also be applied to other modalities.

The learned compatible family’s usefulness is beyond item recommendation. We design a compatible image generator, which can be trained with only the limited labeled data given the succinct representation that has been captured in the compatibility space, bypassing the third challenge. Instead of directly generating the image of a compatible item from a query item, we first obtain a compatible prototype using our system. Then, the prototype is used to generate images of compatible items. This relieves the burden for the generator to simultaneously learn the notion of compatibility and how to generate realistic images. In contrast, existing approaches generate target images directly from source images or source-related features. We propose a novel generator referred to as Metric-regularized Conditional Generative Adversarial Network (MrCGAN). The generator is restricted to work in a similar latent space to the compatibility space. In addition, it learns to avoid generating ambiguous samples that lie on the boundary of two clusters of samples that have conflicting relationships with some query items.

We evaluate our framework on Fashion-MNIST dataset, two Amazon product datasets, and Polyvore outfit dataset. Our method consistently achieves state-of-the-art performance for compatible item recommendation. Finally, we show that we can generate images of compatible items using our learned Compatible Family and MrCGAN. We ask human evaluators to judge the relative compatibility between our generated images and images generated by CGANs conditioned directly on query items. Our generated images are roughly 2x more likely to be voted as compatible.

The main contributions of this paper can be summarized as follows:

- Introduce an end-to-end trainable system for Compatible Family learning to capture asymmetric-relationships.
- Introduce a novel Projected Compatibility Distance to measure compatibility given limited ground truth compatible and incompatible items.
- Propose a Metric-regularized Conditional Generative Adversarial Network model to visually reveal our learned compatible prototypes.

## 2 Related Work

We focus on describing the related work in content-based compatible item recommendation and conditional image generation using Generative Adversarial Networks (GANs).

**Content-based compatible item recommendation** Many works assume a similarity learning setting that requires compatible items to stay close in a learned latent space. McAuley et al. (2015) proposed to use Low-rank Mahalanobis Transform to map compatible items to embeddings close in the latent space. Veit et al. (2015) utilized the co-purchase records from Amazon.com to train a Siamese network to learn representations of items. Lee, Seol, and Lee (2017) assumed that different items in an outfit share a coherent style, and proposed to learn style representations of fashion items by maximizing the probability of item co-occurrences. In contrast, our method is designed to learn asymmetric-relationships.

Several methods go beyond similarity learning. Iwata, Watanabe, and Sawada (2011) proposed to learn a topic model to find compatible tops from bottoms. He, Packer, and McAuley (2016) extended the work of McAuley et al. (2015) to compute “query-item-only” dependent weighted sum (i.e., independent of the compatible items) of  $K$  distances between two items in  $K$  latent spaces to handle heterogeneous item recommendation. However, this means that the query item only prefers several subspaces. While this could deal with diversity across different query items (i.e., different query items have different compatible items), it is less effective for diversity across compatible items of the same query item (i.e., one query item has a diverse set of compatible items). Our model instead represents the distance between an item and a candidate as the minimum of the distance between each prototype and the candidate, allowing compatible items to locate on different locations in the same latent space. In addition, our method is end-to-end trained, and can be coupled with MrCGAN to generate images of compatible items, unlike the other methods.

Another line of research tackles outfit composition as a set problem, and attempts to predict the most compatible item in a multi-item set. Li et al. (2017) proposed learning the representation of an outfit by pooling the item features produced by a multi-modal fusion model. Prediction is made by computing a compatibility score for the outfit representation. Han et al. (2017) treated items in an outfit as a sequence and modeled the item recommendation with a bidirectional LSTM to predict the next item from current ones. Note that these methods require multi-item sets to be given. This potentially can be a limitation in practical use.

**Generative Adversarial Networks** After the introduction of Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) as a popular way to train generative models, GANs have been shown to be capable of conditional generation (Mirza and Osindero 2014), which has wide applications such as image generation by class labels (Odena, Olah, and Shlens 2017) or by texts (Reed et al. 2016), and image transformation between different domains (Isola et al. 2016; Kim et al. 2017; Zhu et al. 2017). Most works for conditional generation focused on similarity relationships. However, compatibility is represented by complex many-to-many relationships, such that the compatible item could be visually distinct from the query item and different query items could have overlapping compatible items.

The idea to regularize GANs with a metric is related to the reconstruction loss used in autoencoders that requires the reconstructed image to stay close to the original image, and it was also applied to GANs by comparing the visual features extracted from samples to enforce the metric in the sample space (Boesen Lindbo Larsen et al. 2015; Che et al. 2016). Our model instead regularizes the subspace of the latent space (*i.e.*, the input of the generator). The subspace does not have the restriction to have known distribution that could be sampled, and visually distinct samples are allowed to be close as long as they have similar compatibility relationships with other items. This allows the subspace to be learned from a far more powerful architecture.

### 3 Our Method

We first introduce a novel Projected Compatibility Distance (PCD) function. Then, we introduce our model architecture and learning objectives. Finally, we introduce a novel metric-regularized Conditional GAN (MrCGAN).

#### 3.1 Projected Compatibility Distance

PCD is proposed to measure the compatibility between two items  $x$  and  $y$ . Each item is transformed into a latent vector by an encoding function  $E_0(\cdot)$ . Additionally,  $K$  projections, denoted as  $\{E_k(\cdot)\}_{k \in \{1, \dots, K\}}$ , are learned to directly map an item to  $K$  latent vectors (*i.e.*, prototypes) close to clusters of its compatible items. Each of the latent vectors has a size of  $N$ . Finally the compatibility distance is computed as follows,

$$d(x, y) = \left\| \frac{\sum_{k=1}^K [\exp(-d_k(x, y)) \mathbf{e}_k^x]}{\sum_{k=1}^K \exp(-d_k(x, y))} - \mathbf{e}_0^y \right\|_2^2 \quad (1)$$

where

$$d_k(x, y) = \|\mathbf{e}_k^x - \mathbf{e}_0^y\|_2^2, \quad (2)$$

and  $\mathbf{e}_k^x$  stands for  $E_k(x)$  for readability.

The concept of PCD is illustrated in Figure 2. When at least one  $\mathbf{e}_k^x$  is close enough to a latent vector  $\mathbf{e}_0^y$  of item  $y$ , the distance  $d(x, y)$  approaches  $\min_k d_k(x, y)$ .

#### 3.2 Model Architecture

In our experiments, most of the layers between  $E_k(\cdot)$  are shared and only the last layers for outputs are separated. This is achieved by using Siamese CNNs (Hadsell, Chopra,

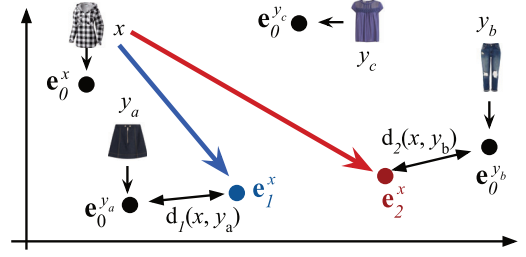


Figure 2: Projected Compatibility Distance. A query item  $x$  is respectively projected to  $\mathbf{e}_1^x$  and  $\mathbf{e}_2^x$  for its two distinct compatible items,  $y_a$  and  $y_b$ . Thus,  $d(x, y_a) \sim d_1(x, y_a)$  for  $y_a$  while  $d(x, y_b) \sim d_2(x, y_b)$  for  $y_b$ . For an incompatible item  $y_c$ , none of the projections are close to  $\mathbf{e}_0^{y_c}$ .

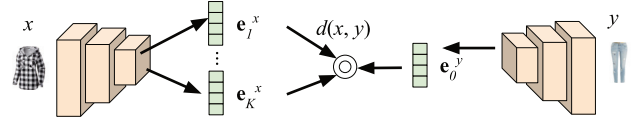


Figure 3: Model Architecture. The CNNs on the left and right are identical. Only the prototypes  $\{\mathbf{e}_k^x\}_{k \in \{1, \dots, K\}}$  from  $x$  and  $\mathbf{e}_0^y$  from  $y$  are considered to form (1).

and LeCun 2006) for feature transformation. As illustrated in Figure 3, rather than learning just an embedding like the original formulation,  $K+1$  projections are learned for the item embedding  $E_0(\cdot)$  and  $K$  prototype projections.

We model the probability of a pair of items  $(x, y)$  being compatible with a shifted-sigmoid function similar to (He, Packer, and McAuley 2016) as shown below,

$$P(x, y) = \sigma_c(-d(x, y)) = \frac{1}{1 + \exp(d(x, y) - c)}, \quad (3)$$

where  $c$  is the shift parameter to be learned.

**Learning objective** Given the set of compatible pairs  $R^+$  and the set of incompatible pairs  $R^-$ , we compute the binary cross-entropy loss, *i.e.*,

$$L_{ce} = -\frac{1}{|R^+|} \sum_{(x, y) \in R^+} \log(P(x, y)) - \frac{1}{|R^-|} \sum_{(x, y) \in R^-} \log(1 - P(x, y)), \quad (4)$$

where  $|\cdot|$  denotes the size of a set. We further regularize the compatibility space by minimizing the distance for compatible pairs. The total loss is as follows:

$$L = L_{ce} + \lambda_m \frac{1}{|R^+|} \sum_{(x, y) \in R^+} d(x, y), \quad (5)$$

where  $\lambda_m$  is a hyper-parameter to balance loss terms.

#### 3.3 MrCGAN for Compatible Item Generation

Once the compatibility space is learned, the metric within the space could be used to regularize a CGAN. The proposed

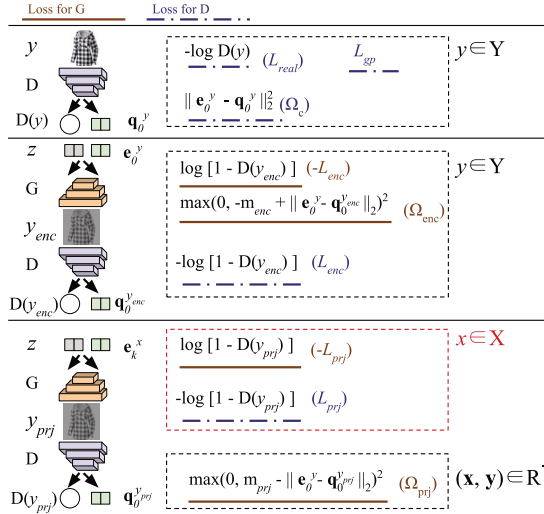


Figure 4: The training procedure of MrCGAN. The generated samples conditioned on different latent vectors are denoted as:  $y_{enc} = G(z, \mathbf{e}_0^y)$ , and  $y_{prj} = G(z, \mathbf{e}_k^x)$ .

model is called Metric-regularized CGAN (MrCGAN). The learning objectives are illustrated in Figure 4.

The discriminator has two outputs:  $D(y)$ , the probability of the sample  $y$  being real, and  $Q_0(y) = \mathbf{q}_0^y$ , the predicted latent vector of  $y$ . The generator is conditioned on both  $z$  and a latent vector from the compatibility space constructed by  $E_0(\cdot)$ . Given the set of query items  $X$ , items to be recommended  $Y$ , and the noise input  $z \in \mathbb{R}^Z \sim p_z = \mathcal{N}(0, 1)$ , we compute the MrCGAN losses as,

$$L_{real} = -\frac{1}{|Y|} \sum_{y \in Y} \log D(y), \quad (6)$$

$$L_{enc} = -\frac{1}{|Y|} \sum_{y \in Y} \mathbb{E}_{z \sim p_z} \log (1 - D(G(z, \mathbf{e}_0^y))), \quad (7)$$

$$L_{prj} = -\frac{1}{|K||X|} \sum_{k=1}^K \sum_{x \in X} \mathbb{E}_{z \sim p_z} \log (1 - D(G(z, \mathbf{e}_k^x))) \quad (8)$$

The discriminator learns to discriminate between real and generated images, while the generator learns to fool the discriminator with both the encoded vector  $\mathbf{e}_0^y$  and the projected prototypes  $\mathbf{e}_k^x$  as conditioned vectors. We also adopt the gradient penalty loss  $L_{gp}$  of DRAGAN (Kodali et al. 2017),

$$L_{gp} = \lambda_{gp} \mathbb{E}_{\hat{y} \sim p_{\text{perturbed } Y}} (\|\nabla_{\hat{y}} D(\hat{y})\|_2 - 1)^2, \quad (9)$$

where a perturbed batch is computed from a batch sampled from  $Y$  as  $\text{batch} + \lambda_{dra} \cdot \text{batch.stddev()} \cdot \mathcal{U}[0, 1]$ , and  $\lambda_{gp}$ ,  $\lambda_{dra}$  are hyper-parameters. In addition, MrCGAN has the following metric regularizers,

$$\Omega_c = \frac{1}{|Y|} \sum_{y \in Y} \|\mathbf{e}_0^y - \mathbf{q}_0^y\|_2^2, \quad (10)$$

which requires the predicted  $\mathbf{q}_0^y$  to stay close to the real latent vector  $\mathbf{e}_0^y$ , forcing  $Q_0(\cdot)$  to approximate  $E_0(\cdot)$ , and

$$\Omega_{enc} = \frac{1}{|Y|} \sum_{y \in Y} M^+(\mathbf{e}_0^y, \mathbf{e}_0^y), \quad (11)$$

where  $m_{enc}$  is a hyper-parameter and

$$M^+(v, s) = \mathbb{E}_{z \sim p_z} \max(0, -m_{enc} + \|v - Q_0(G(z, s))\|_2)^2,$$

which measures the distance between a given vector  $v$  and the predicted latent vector of the generated sample conditioned on  $s$ , and it guides the generator to learn to align its latent space with the compatibility space. The margin  $m_{enc}$  relaxes the constraint so that the generator does not collapse into a 1-to-1 decoder. Finally the generator also learns to avoid generating incompatible items by,

$$\Omega_{prj} = \frac{1}{|K||R^-|} \sum_{k=1}^K \sum_{(x, y) \in R^-} M^-(\mathbf{e}_0^y, \mathbf{e}_k^x), \quad (12)$$

where  $m_{prj}$  is a hyper-parameter and

$$M^-(v, s) = \mathbb{E}_{z \sim p_z} \max(0, m_{prj} - \|v - Q_0(G(z, s))\|_2)^2,$$

which requires the distance between a given latent vector  $v$  and the predicted latent vector of the generated sample conditioned on  $s$  to be larger than a margin  $m_{prj}$ .

The total losses for  $G$  and  $D$  are as below,

$$L_G = -\frac{1}{2}(L_{enc} + L_{prj}) + \Omega_{enc} + \Omega_{prj} \quad (13)$$

$$L_D = L_{real} + \frac{1}{2}(L_{enc} + L_{prj}) + L_{gp} + \Omega_c \quad (14)$$

In effect, the learned latent space is constructed by: (1)  $z$  space, (2) a subspace that has similar structure as the compatibility space. To generate compatible items for  $x$ ,  $\mathbf{e}_k^x$  is used as the conditioning vector:  $G(z, \mathbf{e}_k^x)$ . To generate items with similar style to  $y$ ,  $\mathbf{e}_0^y$  is used instead:  $G(z, \mathbf{e}_0^y)$ .

### 3.4 Implementation Details

We set  $\lambda_m$  to 0 and 0.5 respectively for recommendation and generation experiments and the batch size to 100, and use Adam optimizer with  $(\lambda_{lr}, \beta_1, \beta_2) = (0.001, 0.9, 0.999)$ . The validation set is for the best epoch selection. The last layer of each discriminative model takes a fully-connected layer with weight normalization (Salimans and Kingma 2016) except for the Amazon also-bought/viewed experiments, where the weight normalization is not used for fair comparison. Before the last layer, the following feature extractors are for different experiments: (1) Fashion-MNIST+1+2 / MNIST+1+2: multi-layer CNNs with weight normalization, (2) Amazon also-bought/viewed: none, (3) Amazon co-purchase: Inception-V1 (Szegedy et al. 2015), (4) Polyvore: Inception-V3 (Szegedy et al. 2016).

For the generation experiments, we set  $\lambda_{gp}$  to 0 and apply DCGAN (Radford, Metz, and Chintala 2015) architecture for both of our model and the GAN-INT-CLS (Reed



et al. 2016) in MNIST+1+2. For Amazon co-purchase and Polyvore generation experiments, we set both  $\lambda_{gp}$  and  $\lambda_{dra}$  to 0.5, and adopt a SRResNet(Ledig et al. 2016)-like architecture for GAN training with a different learning rate setting, *i.e.*,  $(\lambda_{lr}, \beta_1, \beta_2) = (0.0002, 0.5, 0.999)$ . The model and parameter choosing is inspired by Jin et al. (2017), but we take off the skip connections from the generator since it does not work well in our experiment, and we also use weight normalization for all layers.<sup>1</sup>

In most of our experiments, the sets,  $X$  and  $Y$ , are identical. However, we create non-overlapped sets of  $X$  and  $Y$  by restricting the categories in the generation experiments for Amazon co-purchase and Polyvore dataset. The dimension of  $z$  and the number of  $K$  are respectively set to 20 and 2 in all generation experiments. Besides, the rest of the parameters are taken as follows: (1) MNIST+1+2:  $(N, m_{enc}, m_{prj}) = (20, 0.1, 0.5)$ , (2) Amazon co-purchase:  $(N, m_{enc}, m_{prj}) = (64, 0.05, 0.2)$ , (3) Polyvore:  $(N, m_{enc}, m_{prj}) = (20, 0.05, 0.3)$ .

## 4 Experiment

We conduct experiments on several datasets and compare the performance with state-of-the-art methods for both compatible item recommendation and generation.

### 4.1 Recommendation Experiments

**Baseline** Our proposed PCD is compared with two baselines: (1) the  $L_2$  distance between the latent vectors of the Siamese model, (2) the Monomer proposed by He, Packer, and McAuley (2016). Although Monomer was originally not trained end-to-end, we still cast an end-to-end setting for it on Fashion-MNIST+1+2 dataset.

Our model achieves superior performance in most experiments. In addition, our model has two advantages in efficiency compared to Monomer: (1) our storage is  $\frac{1}{K}$  of Monomer since they projected each item into  $K$  spaces beforehand while we only compute  $K$  prototype projections during recommendation, (2) PCD is approximately  $\min_k d_k(x, y)$ . Therefore, in query time we could do  $K$  nearest-neighbor search in parallel to get approximate results, while Monomer needs to aggregate the weighted sum of the distances in  $K$  latent spaces.

**Fashion-MNIST+1+2 Dataset** To show our model’s ability to handle asymmetric relationships, we build a toy dataset from Fashion-MNIST (Xiao, Rasul, and Vollgraf 2017). The dataset consists of 28x28 gray-scale images of 10 classes, including T-shirt, Trouser, Pullover, etc. We create an arbitrary asymmetric relationship as follows,

$$(x, y) \in R^+ \iff C_y = (C_x + i) \mod 10, \forall i \in \{1, 2\},$$

where  $C_x$  means the class of  $x$ . The other cases of  $(x, y)$  belong to  $R^-$ .

Among the 60,000 training samples, 16,500 and 1,500 pairs are non-overlapped and randomly selected to form the

<sup>1</sup>The architecture for  $G$  is FC4x4x64-CONV2048-CONV1024-CONV512-CONV12-Tanh, where Pixel Shufflers and ReLUs are used in each layer, and  $kernel = 3$ ,  $stride = 1$ .

Model	K	Error rate	AUC
$L_2$		41.45% $\pm$ 0.55%	0.6178 $\pm$ 0.0034
Monomer	2	23.24% $\pm$ 0.62%	0.8533 $\pm$ 0.0045
	3	22.46% $\pm$ 0.44%	0.8598 $\pm$ 0.0021
	4	21.60% $\pm$ 0.39%	0.8651 $\pm$ 0.0052
	5	22.03% $\pm$ 0.39%	0.8623 $\pm$ 0.0030
PCD	2	21.31% $\pm$ 0.77%	0.8746 $\pm$ 0.0070
	3	20.66% $\pm$ 0.30%	0.8804 $\pm$ 0.0033
	4	20.39% $\pm$ 0.60%	0.8808 $\pm$ 0.0043
	5	<b>20.01%</b> $\pm$ 0.39%	<b>0.8830</b> $\pm$ 0.0032

Table 1: Performance on Fashion-MNIST+1+2.

Dataset	Graph	LMT	Monomer	PCD
Men	also_bought	9.20%	6.48%	<b>6.05%</b>
	also_viewed	6.78%	6.58%	<b>5.97%</b>
Women	also_bought	11.52%	7.87%	<b>7.75%</b>
	also_viewed	7.90%	<b>7.34%</b>	7.37%
Boys	also_bought	8.80%	5.71%	<b>5.27%</b>
	also_viewed	6.72%	5.35%	<b>5.03%</b>
Girls	also_bought	8.33%	5.78%	<b>5.34%</b>
	also_viewed	6.46%	5.62%	<b>4.86%</b>
Baby	also_bought	12.48%	7.94%	<b>7.00%</b>
	also_viewed	11.88%	9.25%	<b>8.00%</b>
Avg.		9.00%	6.79%	<b>6.26%</b>

Table 2: Error rates on Amazon also-bought/viewed dataset. LMT stands for Low-rank Mahalanobis Transform (McAuley et al. 2015).

training and validation sets, respectively. Besides, 10,000 pairs are created from the 10,000 testing samples for the testing set. The samples in each split are also non-overlapped. The strategy of forming a pair is that we randomly choose a negative or a positive sample  $y$  for each sample  $x$ . Thus,  $|R^+| \approx |R^-|$ . We erase the class labels while only keep the pair labels during training for the reason that the underlying factor for compatibility is generally unavailable.

We repeat the above setting five times and show the averaged result in Table 1. Like the settings of He, Packer, and McAuley (2016), the latent size in  $L_2$  equals to  $(K+1) \times N$ . Here, the size in  $L_2$  is 60. Each model is trained for 50 epochs. The experiment shows that the  $L_2$  model performs poorly on a highly asymmetric dataset while our model achieves the best results.

**Amazon also-bought/viewed Dataset** The image features of 4096 dimensions are extracted beforehand in this dataset. Following the setting of He, Packer, and McAuley (2016), the also-bought and also-viewed relationships in the Amazon dataset (McAuley et al. 2015) are positive pairs while the negative pairs are sampled accordingly. We set our parameters,  $K$  and  $N$ , as the same as Monomer for comparison, *i.e.*, (1)  $(K, N) = (4, 20)$  for also-bought, and (2)  $(K, N) = (4, 10)$  for also-viewed. We train 200 epochs for each model and Table 2 shows the results. Compared to the error rates reported in He, Packer, and McAuley (2016), our model yields the best performance in most settings and the

Model	AUC
Veit et al. (2015)	0.826
Veit et al. (2015) (retrain last layer)	0.8698
Monomer	0.8747
PCD	<b>0.8762</b>

Table 3: AUC on Amazon co-purchase dataset.

lowest error rate on average.

**Amazon Co-purchase Dataset** Based on the data split<sup>2,3</sup> defined in Veit et al. (2015), we increase the validation set via randomly selecting an additional 9,996 pairs from the original training set since its original size is too small, and accordingly decrease the training set by removing the related pairs for the non-overlapping requirement. Totally, 1,824,974 pairs remain in the training set. As the ratio of positive and negative pairs is disrupted, we re-weight each sample during training to re-balance it back to 1:16. Besides, we randomly take one direction for each pair since the ‘‘Co-purchase’’ relation in this dataset is symmetric.

We adopt and fix the pre-trained weights from Veit et al. (2015), and replace the last embedding layer with each model’s projection layer. The last layer is trained for 5 epochs in each model. Still, we set the latent size in Veit et al. (2015) as 256 and equal to  $(K+1) \times N$ . Thus,  $(K, N) = (3, 64)$  for both Monomer and our model. Table 3 shows the results. Our model obtains superior performance to Veit et al. (2015) and comparable results with Monomer.

**Polyvore Dataset** To demonstrate the ability of our model to capture the implicit nature of compatibility, we create an outfit dataset from Polyvore.com, a collection of user-generated fashion outfits. We crawl outfits under Women’s Fashion category and group the category of items into tops, bottoms, and shoes. Outfits are ranked by the number of likes and the top 20% are chosen as positive.

Three datasets are constructed for different recommendation settings: (1) from tops to others, (2) from bottoms to others, and (3) from shoes to others. The construction procedure is as follows: (1) Items of source and target categories are non-overlapped split according to the ratios 60 : 20 : 20 for training, validation, and test sets. (2) A positive pair is decided if it belongs to the positive outfit. (3) The other combinations are negative and sub-sampled by choosing 4 items from target categories for each positive pair. Duplicate pairs are dropped afterwards. This dataset is more difficult since the compatibility information across categories no longer exists, *i.e.*, both positive and negative pairs are from the same categories. The model is forced to learn the elusive compatibility between items.

Pre-trained Inception-V3 is used to extract image features, and the last layer is trained for 50 epochs. We set  $N$  to 100 for  $L_2$ , and  $(K, N) = (4, 20)$  for Monomer and PCD. The AUC scores are listed in Table 4, and our model still

<sup>2</sup><https://vision.cornell.edu/se3/projects/clothing-style/>

<sup>3</sup>7 images from the training set are missing, so 35 pairs containing these images are dropped.

Graph	$L_2$	Monomer	PCD
top-to-others	0.7165	0.7431	<b>0.7484</b>
shoe-to-others	0.6988	0.7146	<b>0.7165</b>
bottom-to-others	0.7450	0.7637	<b>0.7680</b>
Avg.	0.7201	0.7405	<b>0.7443</b>

Table 4: AUC on Polyvore dataset.



Figure 5: Column  $x$  is the input. The rows show the generated images by varying  $z$  and  $k$  of different methods.

achieves the best results.

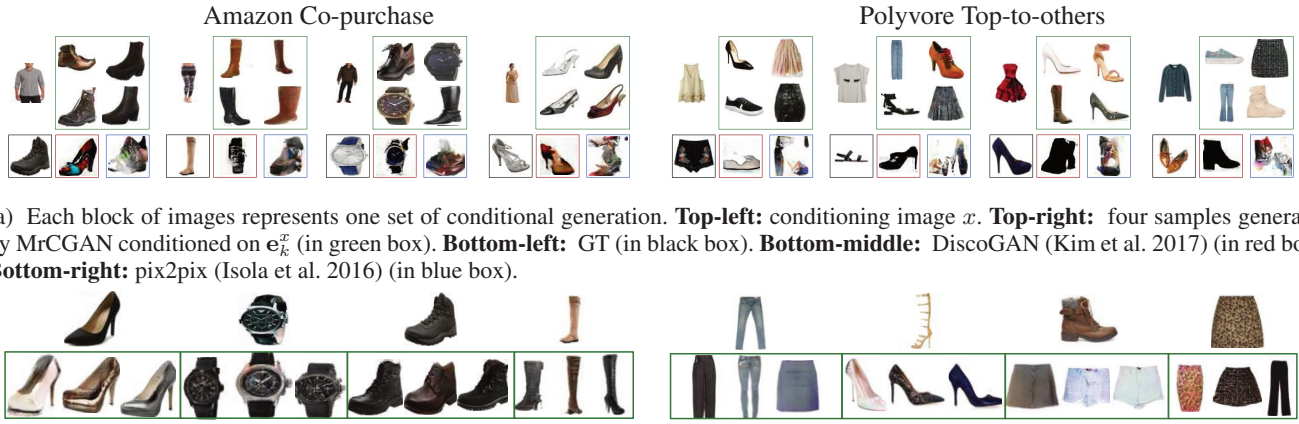
## 4.2 Generation Experiments

**Baseline** We compare our model with GAN-INT-CLS from (Reed et al. 2016) in the MNIST+1+2 experiment.  $e_0^x$  is used as the conditioning vector for GAN-INT-CLS, and each model uses exact architecture except that the discriminator for MrCGAN outputs an additional  $Q_0(\cdot)$ , while the discriminator for GAN-INT-CLS is conditioned by  $e_0^x$ . For other experiments, we compare with pix2pix (Isola et al. 2016) that utilizes labeled image pairs for image-to-image translation and DiscoGAN (Kim et al. 2017) that unsupervisedly learns to transform images into a different domain.

**MNIST+1+2** We use MNIST dataset to build a similar dataset as Fashion-MNIST+1+2 because the result is easier to interpret. Additional 38,500 samples are selected as unlabeled data to train the GANs. In Figure 5 we display the generation results conditioned on samples from the test set. We found that our model could preserve diversity better and that the  $K$  projections automatically group different modes, so the variation can be controlled by changing  $k$ .

**Amazon Co-purchase Dataset** We sample a subset from Amazon Co-purchase dataset (Veit et al. 2015) by reducing the number of types for target items so that it’s easier for GANs to work with. In particular, we keep only relationships from Clothing to Watches and Shoes in Women’s and Men’s categories. We re-weight each sample during training to balance the ratio between positives and negatives to 1:1, but for validation set, we simply drop all excessive negative pairs. Totally, there remain 9,176 : 86 : 557 positive pairs and 435,396 : 86 : 22,521 negative pairs for training, validation and test split, respectively. The unlabeled training set for GANs is selected from the training ids from Veit et al. (2015), and it consist of 226,566 and 252,476 items of source and target categories, respectively. Each image is also resized to 64x64 before being fed into the discriminator.

Both DiscoGAN and pix2pix are inherently designed for similarity learning, and as shown in Figure 6a, they could



(a) Each block of images represents one set of conditional generation. **Top-left:** conditioning image  $x$ . **Top-right:** four samples generated by MrCGAN conditioned on  $e_k^x$  (in green box). **Bottom-left:** GT (in black box). **Bottom-middle:** DiscoGAN (Kim et al. 2017) (in red box). **Bottom-right:** pix2pix (Isola et al. 2016) (in blue box).

(b) Each block of images represents conditional generation (in green box) based on the latent vector  $e_0^y$  of the image  $y$  on the top. Note that by conditioning on  $e_0^y$ , MrCGAN generates items having similar style as  $y$  instead of compatible items.

Figure 6: Examples of generated images conditioned on (a)  $e_k^x$  and (b)  $e_0^y$ .

not produce satisfying results. Moreover, our model produces diverse outputs, while the diversity for conventional image-to-image models are limited. We additionally sample images conditioned on  $e_0^y$  instead of on  $e_k^x$  as illustrated in Figure 6b. We find that the MrCGAN has the ability to generate diverse items having similar style as  $y$ .

**Polyvore Top-to-others Dataset** We use the top-to-others split as a demonstration for our methods. Likewise we reweigh each sample during training to balance positive and negative pairs to 1:1 to encourage the model to focus on positive pairs. The results are shown in Figure 6.

**User study** Finally we conduct online user surveys to see whether our model could produce images that are perceived as compatible. We conduct two types of surveys: (1) Users are given a random image from source categories and three generated images by different models in a randomized order. Users are asked to select the item which is most compatible with the source item, and if none of the items are compatible, select the most recognizable one. (2) Users are given a random image from source categories, a random image from target categories, and an image generated by MrCGAN in a randomized order. Users are asked to select the most compatible item with the source item, and if none of the items are compatible, users can decline to answer. The results are shown in Figure 7, and it shows that MrCGAN can generate compatible and realistic images under compatibility learning setting compared to baselines. While the difference against random images is small in the Polyvore survey, MrCGAN is significantly preferred in the Amazon co-purchase survey. This is also consistent with the discriminative performance.

### 4.3 Discussion

As shown in Table 1, a larger  $K$  gives better performance when the total embedding dimension  $(K+1) \times N$  is kept equal, but the differences are small if the size is increased continuously. The regularizer controlled by  $\lambda_m$  forces the distances between prototypes and a compatible item to be

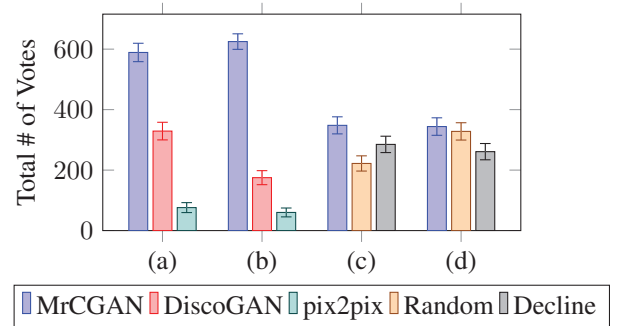


Figure 7: Survey results on Amazon co-purchase (see panel (a, c)) and Polyvore (see panel (b, d)). MrCGAN (blue) generally outperforms others.

small. We found that this constraint hurts recommendation performance, but it could be helpful for generation. We recommend choosing a smaller  $\lambda_m$  when the quality difference between images generated from  $e_k^x$  and  $e_0^y$  is not noticeable.

From the four datasets, the recommendation performance of our model is gradually getting closer to the others, which might be due to the disappearance of the asymmetric relationship. In Fashion-MNIST+1+2, this relationship is injected by force. Then Amazon also-bought/viewed dataset preserves buying and viewing orders so some asymmetric relationship exists. However, only the symmetric relationship remains for Amazon co-purchase and Polyvore datasets. This suggests our model is suitable for asymmetric relationship but it still works well under symmetric settings.

We found that a larger  $m_{prj}$  reduced the diversity but a smaller  $m_{prj}$  made incompatible items more likely to appear. In practice, it works well to set  $m_{prj}$  to be slightly larger than the average distances of positive pairs from the training set. Removing margin  $m_{enc}$  seems to decrease the diversity in simple datasets such as MNIST+1+2, but we did not tune it on other complex datasets.



## 5 Conclusion

We propose modeling the asymmetric and many-to-many relationship of compatibility by learning a Compatibility Family of representation and prototypes with an end-to-end system and the novel Projected Compatible Distance function. The learned Compatibility Family achieves more accurate recommendation results when compared with the state-of-the-art Monomer method (He, Packer, and McAuley 2016) on real-world datasets. Furthermore, the learned Compatibility Family resides in a meaningful Compatibility Space and can be seamlessly coupled with our proposed MrCGAN model to generate images of compatible items. The generated images validate the capability of the Compatibility Family in modeling many-to-many relationships. Furthermore, when compared with other approaches for generating compatible images, the proposed MrCGAN model is significantly more preferred in our user surveys. The recommendation and generation results justify the usefulness of the learned Compatibility Family.

## Acknowledgement

The authors thank Chih-Han Yu and other colleagues of Appier as well as the anonymous reviewers for their constructive comments. We thank Chia-Yu Joey Tung for organizing the user study. The work was mostly completed during Min Sun’s visit to Appier for summer research, and was part of the industrial collaboration project between National Tsing Hua University and Appier. We also thank the supports from MOST project 106-2221-E-007-107.

## References

- Boesen Lindbo Larsen, A.; Kaae Sønderby, S.; Larochelle, H.; and Winther, O. 2015. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*.
- Che, T.; Li, Y.; Jacob, A. P.; Bengio, Y.; and Li, W. 2016. Mode Regularized Generative Adversarial Networks. *arXiv preprint arXiv:1612.02136*.
- Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc. 2672–2680.
- Hadsell, R.; Chopra, S.; and LeCun, Y. 2006. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR ’06, 1735–1742. Washington, DC, USA: IEEE Computer Society.
- Han, X.; Wu, Z.; Jiang, Y.-G.; and Davis, L. S. 2017. Learning fashion compatibility with bidirectional LSTMs. In *Proceedings of the 2017 ACM on Multimedia Conference*, 1078–1086. ACM.
- He, R.; Packer, C.; and McAuley, J. 2016. Learning compatibility across categories for heterogeneous item recommendation. In *International Conference on Data Mining*, 937–942. IEEE.
- Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2016. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*.
- Iwata, T.; Watanabe, S.; and Sawada, H. 2011. Fashion coordinates recommender system using photographs from fashion magazines. In *International Joint Conference on Artificial Intelligence*, 2262–2267. AAAI Press.
- Jin, Y.; Zhang, J.; Li, M.; Tian, Y.; Zhu, H.; and Fang, Z. 2017. Towards the Automatic Anime Characters Creation with Generative Adversarial Networks. *arXiv preprint arXiv:1708.05509*.
- Kim, T.; Cha, M.; Kim, H.; Lee, J. K.; and Kim, J. 2017. Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, 1857–1865. PMLR.
- Kodali, N.; Abernethy, J.; Hays, J.; and Kira, Z. 2017. How to Train Your DRAGAN. *arXiv preprint arXiv:1705.07215*.
- Ledig, C.; Theis, L.; Huszar, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; and Shi, W. 2016. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *arXiv preprint arXiv:1609.04802*.
- Lee, H.; Seol, J.; and Lee, S.-g. 2017. Style2Vec: Representation Learning for Fashion Items from Style Sets. *arXiv preprint arXiv:1708.04014*.
- Li, Y.; Cao, L.; Zhu, J.; and Luo, J. 2017. Mining fashion outfit composition using an end-to-end deep learning approach on set data. *IEEE Trans. Multimedia* 19:1946–1955.
- McAuley, J. J.; Targett, C.; Shi, Q.; and van den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*, 43–52. ACM.
- Mirza, M., and Osindero, S. 2014. Conditional generative adversarial nets. *CoRR* abs/1411.1784.
- Odena, A.; Olah, C.; and Shlens, J. 2017. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the 34th International Conference on Machine Learning*, 2642–2651. PMLR.
- Radford, A.; Metz, L.; and Chintala, S. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*.
- Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; and Lee, H. 2016. Generative adversarial text-to-image synthesis. In *Proceedings of The 33rd International Conference on Machine Learning*, 1060–1069. PMLR.
- Salimans, T., and Kingma, D. P. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc. 901–909.
- Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.
- Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.
- Veit, A.; Kovacs, B.; Bell, S.; McAuley, J.; Bala, K.; and Belongie, S. 2015. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*, 4642–4650.
- Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*.
- Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*.