Using Discriminative Methods to Learn Fashion Compatibility Across Datasets

Kedan Li, Chen Liu, Ranjitha Kumar, and David Forsyth
Department of Computer Science
University of Illinois at Urbana-Champaign
{kedanli2, chenliu8, ranjitha, daf}@illinois.edu

Abstract

Determining whether a pair of garments are compatible with each other is a challenging matching problem. Past works explored various embedding methods for learning such a relationship. This paper introduces using discriminative methods to learn compatibility, by formulating the task as a simple binary classification problem. We evaluate our approach using an established dataset of outfits created by non-experts and demonstrated an improvement of 2.5% on established metrics over the state-of-the-art method. We introduce three new datasets of professionally curated outfits and show the consistent performance of our approach on expert-curated datasets. To facilitate comparing across outfit datasets, we propose a new metric which, unlike previously used metrics, is not biased by the average size of outfits. We also demonstrate that compatibility between two types of items can be query indirectly, and such query strategy yield improvements.

1 Introduction

Predicting which fashion items will go together to form an outfit is a high-value task in e-commerce. While there is an established literature on this topic, it remains hard to build accurate systems. The key is to tell whether two items are compatible, which is complicated because both complex appearance properties and perceptual issues are in play. Current methods rely on similarity measures obtained by learning embeddings. Until recently, such approaches have ignored garment type (e.g. "dress" vs. "hat"), which is odd because one does not usually have many items of the same kind in an outfit Veit* et al. (2015); He et al. (2016). Recent work has shown that acknowledging type produces improvements in standard metrics Vasileva et al. (2018).

Remarkably, we are not aware of any recent work using discriminative method to predict compatibility. This absence is odd because the problem is naturally discriminative — one wants to know whether a particular pair of items is compatible or not. Embedding methods try to solve a harder problem of learning a good metric of similarity. In this paper, we demonstrate that discriminative methods produce robust systems. Our methods exceed the state-of-the-art on established metrics by 2.5% on both compatibility prediction task and fill-in-the-blank task on the Polyvore Outfits dataset.

The current standard, Polyvore Outfits, was created by online community users who are not necessarily experts in the fashion domain. To learn the compatibility rules defined by professionals, we introduce three new outfit datasets from crawling three e-commerce sites consisting of 360,176 outfits in total. We evaluate our discriminative method across datasets and demonstrate that the method performs consistently well on expert-curated outfit data. We also introduce pairwise AUC as a new metric for measuring compatibility. Unlike compatibility (per outfit) AUC used by prior works, pairwise AUC is not misled by the average size of outfits in the dataset when comparing across datasets.

Preprint. Under review.

To compare the compatibility rules defined by multiple sources, we compute the type-pair AUCs for every pair of product types (e.g., outerwear-bottom) and visualize them using heat maps. The analysis reveals that the underlying ways items match in different outfit datasets are different, which suggests that the notion of fashion compatibility is subjective. The type-pair AUCs also indicates that the model predicts some type-pairs more accurately than others. Exploring this property, we show that compatibility can also be queried indirectly, using a third type as the anchor. We demonstrate that indirect query yield improvement in the AUC results for the infrequent pairs of types.

Our contributions are:

- **Three new datasets:** We introduce three new datasets generated by professional stylists from three e-commerce sites, consisting of 360,176 outfits in total.
- **Improved accuracy:** We show that a discriminative approach improves accuracy over state of the art in learning compatibility relationship.
- **Cross-dataset comparisons:** We show that the method performs consistently well on the standard dataset and our three datasets. However, the current evaluation uses compatibility (per outfit) AUC. We show that, for statistical reasons, compatibility AUC is higher on datasets with larger outfits. We describe pairwise AUC as an alternative metric for measuring compatibility performance and demonstrate that it is more stable when comparing across outfit datasets.
- **Indirect query strategy:** The model performs poorly for pairs of types with relatively few compatible examples in the training data. We show that, for these pairs, an indirect strategy for querying compatibility (e.g., searching for "eyewear" that are compatible with a "scarf" through "tops" that are compatible with both) improves over a direct query.

2 Related Work

The established approach for learning representations of complex relationships involves constructing an embedding space by training as a siamese structure Chopra et al. (2005) or using triplet loss Schroff et al. (2015) with samples of positive and negative pairs. Embedding methods are also commonly used to capture hard-to-define relationships in the fashion domain such as style, fashionability, and the matching between clothing items McAuley et al. (2015a); Simo-Serra and Ishikawa (2016); Hsiao and Grauman (2017); Simo-Serra et al. (2015); Veit* et al. (2015); He et al. (2016). Han et al. (2017) trained a bidirectional LSTM model to predict the next compatible clothing item within an outfit, always regarding an outfit as a whole. Noticed the non-transitive nature of the compatibility relationship, Vasileva et al. (2018) recently demonstrated that enforcing type-awareness in the embeddings produce better performance over prior works.

Discriminative methods, or in other words classification methods, have been commonly used to solve a variety of standard computer vision tasks, such as image classification, object detection, image segmentation Krizhevsky et al. (2012); Ren et al. (2015); Ronneberger et al. (2015), etc. However, surprisingly, we are unable to find examples of its usage in solving the compatibility problem.

Prior works mainly used curated outfits as supervision signals and the ground truth for evaluation to learn fashion compatibility. Many recent works, including the state-of-the-art, are trained and evaluated using outfits data mined from Polyvore Han et al. (2017); Vasileva et al. (2018). Polyvore is a social network where fashion lovers curate outfits using a set of fashion product images. We emphasize that the quality of the outfits curated by Polyvore users are not guaranteed. Also, prior works only evaluate their methods using outfits from a single source Han et al. (2017); Vasileva et al. (2018). Assessing one outfit dataset does not ensure a method's performance on other outfit datasets.

In the fashion domain, other recent works focused on products recognition and retrieval Liu et al. (2012); Yang et al. (2014); Hu et al. (2015); Kiapour et al. (2015), fashion recommendation Vittayakorn et al. (2016); Shen et al. (2007); McAuley et al. (2015b), fashion attributes detections Kiapour et al. (2014) and discovering fashion trends Al-Halah et al. (2017).

Table 1: The table above shows the combined statistics for the three e-commerce datasets. The table below shows the statistics for each of the three datasets.

No. of	No. of	Outfits of k items						
Outfits	Items	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7	k > 7
360,176	636,407	104,144	100,049	74, 813	42, 704	20,702	9,926	7,838

Farfetch		Net-A-Porter		Modaoperandi	
No. of Outfits	No. of Items	No. of Outfits	No. of Items	No. of Outfits	No. of Items
234,591	501,610	88,251	111,249	14,336	46,546

3 E-commerce datasets

To learn compatibility relationships that meet the professional standard, one should use outfits created by experts. As a contribution, we introduce three new datasets consists of outfits made by professional stylists in three fashion e-commerce sites. The E-commerce datasets comprise of 636,407 fashion products from three different e-commerce sites: Farfetch (www.farfetch.com), Net-A-Porter (www.net-a-porter.com) and Moda Operandi (www.modaoperandi.com). For each product, the site provides a studio photo of a model wearing the item together with a set of other products, forming a complete outfit. The websites contain links to other products. We mine the outfit data directly from the e-commerce websites using web-scraping techniques. As the links may disappear when products go out-of-stock, we may not always obtain the complete outfit in the studio image.

We obtained a total of 360,176 unique outfits, each comprises of two or more fashion products, with statistic shown in Table 1. Every fashion item comes with a product image of white background, a name, a list of category keywords, and other metadata. We specially select the three sites because they consistently supply labeled front-view product image with a white background, which match the Polyvore Outfits dataset. To obtain the 11 type labels used by Vasileva et al. (2018), we train a classification model to predict the type from product metadata. The model using the text CNN architecture introduced by Kim (2014). Training labels are obtained by manually create a mapping between the Farfetch product category keywords and the 11 target types. The classification model achieves over 99% accuracy and is used to label the type for all products.

We split the outfit data into three datasets, and each only comprises of outfits from one e-commerce site. The split is necessary because outfits from different sites may not follow the same compatibility rules, which is later verified by the results. Each dataset is split into 60% for training, 20% for validation, and 20% for testing, based on the number of outfits. We do not prevent shared items between different splits, because Vasileva et al. (2018) showed that such operation has minimal effect on the results but significantly reduce the number of available outfits. From the test set, we create tests for the Fashion Compatibility Prediction task introduced by Han et al. (2017). The original outfits are regarded as positive examples. We create a negative example to match every positive example by substitute each item in the outfit by a random item of the same type.

4 Discriminative methods for predicting compatibility

We must learn a function s that takes a pair of fashion item images $(\mathcal{I}_i, \mathcal{I}_j)$ as input, and produces a compatibility score in the range $[0, 1]$. This score will be tested against a threshold, to be chosen later. This function is computed from a (learned) feature embedding vector $\mathbf{e}(\mathcal{I})$ computed from each separate image. From the embedding, we compute a joint feature vector $\mathbf{z} = \mathbf{z}(\mathbf{e}(\mathcal{I}_i), \mathbf{e}(\mathcal{I}_j))$, then apply a predictor to obtain

$$s(\mathcal{I}_i, \mathcal{I}_j) = s(\mathbf{z}(\mathbf{e}(\mathcal{I}_i), \mathbf{e}(\mathcal{I}_j)))$$

Training data consists of pairs that are compatible or incompatible, and the function is learned using binary cross-entropy loss. For the image encoder e, we use the identical architecture used by Vasileva et al. (2018), with ResNet18 as the backbone He et al. (2016). The scoring function s consists of a fully connected neural network with 2 hidden layers with ReLU non-linearity and an output layer with sigmoid activation. We investigate various choices of $\mathbf{z}$, constructed to be symmetric in their arguments. Write z_i for the i'th component of vector $\mathbf{z}$. We consider the following options:

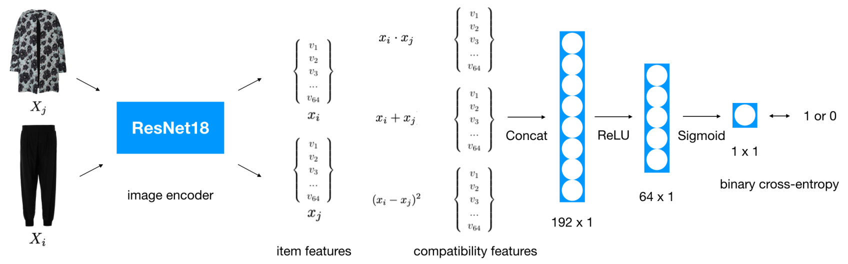

Figure 1: The diagram shows the architecture of our method. The architecture takes in two images of fashion items as input, computes each item's embedding, combines the item embeddings into a compatibility representation, and predict a probability score of the two items being compatible.

- **dot**, where $\mathbf{z}_{dot}(\mathbf{u}, \mathbf{v}) = [u_1 v_1, \ldots, u_d v_d]^T$;
- **diff**, where $\mathbf{z}_{diff}(\mathbf{u}, \mathbf{v}) = \left[(u_1 - v_1)^2, \ldots, (u_d - v_d)^2\right]^T$;
- **sum**, where $\mathbf{z}_{sum}(\mathbf{u}, \mathbf{v}) = \mathbf{u} + \mathbf{v}$.

Each appears to contribute strongly to the predictions. We investigate producing feature vectors by stacking the three options (using all turns out to be best). As each of the formulation predicts compatibility, we argue that supplying all three of them to the probability predictor function will yield better result. We write the concatenation of the feature vectors, for example **dot**+**sum**, as

$$\left[\mathbf{z}_{dot}^T, \mathbf{x}_{sum}^T\right]^T$$

and so on.

A natural generalization is to form an outer product, where

$$\mathbf{z}_{op}(\mathbf{u}, \mathbf{v}) = [2u_1 v_1, \ldots, u_r v_s + v_r u_s, \ldots, u_r, \ldots, v_s, \ldots, 1].$$

While this exposes all possible pairwise quadratic and linear monomials, it does not outperform the other feature vectors (demonstrated by the study) and is inconveniently large.

4.1 Generating the training data

All pairs of items that occur in an outfit are assumed compatible, so we use all pairs in any training outfit as positive training pairs. As is usual for embedding methods, we assume that an arbitrary pair that does not appear in an outfit is incompatible. When randomly sampling negative items, we adopt the category-aware negative sampling method introduced by Vasileva et al. (2018). The method requires the negative item to have the same category as the positive items.

4.2 Experiments

We conduct an experiment to compare the performance of our methods against the state-of-the-art compatibility model on the established Polyvore Outfit dataset. The results demonstrate that our model outperforms the best prior work by 2.5% on AUC and fill-in-the-blank test. An ablation study is also conducted to illustrate different variants of our method.

Metrics Following prior works, we evaluate our method on the fashion compatibility task and the fill-in-the-blank test (FITB) task. Fill in the blank measures a method's accuracy in choosing the correct item from a list of four to fill in a blank in an outfit. For fashion compatibility task, we follow the usual practice of averaging the score for all pairs of items in an outfit, then computing AUC for ground truth outfits against random outfits (compatibility AUC). For both tasks, we use the standard test set created by Vasileva et al. (2018). In section 5, we show that compatibility AUC is inclined to be higher for datasets where outfits are larger, which could mislead. Therefore, we also report the pairwise AUC, defined as the AUC for all pairs (of any type) under the compatibility score.

Table 2: The table compares the performance of the discriminative method and the state-of-the-art method on Polyvore Outfits dataset. n-D stands for the size of the embedding or item feature vector.

Method	compatibility AUC	pairwise AUC	FITB
type-aware embedding (64-D)	.862	.654	55.3%
type-aware embedding (512-D)	.875	.695	58.0%
discriminative method (64-D)	.895	.715	59.1%
discriminative method (512-D)	**.903**	**.720**	**60.4%**

Table 3: The table compares the performance of the discriminative methods using different transformations on Polyvore Outfits dataset. All conditions use item feature vectors of size 64.

Transformation Function	compatibility AUC	pairwise AUC	FITB
diff	.853	.687	52.5%
sum	.858	.691	53.5%
dot	.878	.703	56.1%
outer product	.879	.704	56.2%
diff + sum	.879	.704	56.8%
dot + diff	.889	.711	57.6%
dot + sum	.892	.712	58.6%
dot + diff + sum	**.895**	**.715**	**59.1%**

Experiment Setting To compare these methods, we use the full Polyvore Outfits dataset, as by Vasileva et al. (2018). We compare to the type-aware embedding compatibility model of that paper. This model is advantaged over our methods because it is trained using text information (precomputed HGLMM Fisher vectors from Klein et al. (2014)) as well as image information; our models use only image information. Each method is trained for 20 epochs using Adam optimizers with a learning rate of 10e-5. We choose the epoch with the highest average of pairwise AUC and compatibility AUC on the validation set for testing.

Results Our discriminative method strongly outperforms the type-aware embedding (Table 2), despite not possessing text information. Table 3 shows the performance between different variants of our method on the Polyvore Outfits Dataset. All conditions are trained using item representation of 64, as Table 2 confirmed that using larger item representation yield better results. Using transformation of **dot** alone performs 2% better than **diff** and **sum** alone. Concatenating either **diff** or **sum** with **dot** increase the performance by 1%. Concatenating the three relationships yield the best performance with an additional improvement of 1%. The outer product yields minimal improvement over **dot**, and so can be discarded.

5 Pairwise AUC and multiple datasets

We evaluate our best performing model (concatenating **dot**, **diff** and **sum** features, retrained as appropriate) on the three e-commerce datasets to demonstrate the consistent strength of our approach. However, we carefully choose the metric to use, because different datasets have different average outfit size, as shown in Table 5.

5.1 Pairwise AUC vs. Compatibility AUC

Datasets with a large average outfits size will tend to have higher compatibility AUC, quite independent of the accuracy of the prediction of individual compatibility scores. This effect is easily seen with a simple model. Assume the compatibility predictor produces a score that is a normal random variable. Scale and translate as required so that this score is distributed as $N(0,1)$ for non-compatible edges; in this case, for compatible edges, the score will be distributed as $N(\mu,\sigma)$, for $\mu,\sigma > 0$. Write $A(\mu,\sigma)$ for the AUC computed from these distributions. Then the false positive rate at some threshold t is $f(t) = 1 - \Phi(t)$ (for Φ the cumulative distribution of the unit normal) and the true positive

Table 4: This table shows the result of training and crossly-testing the best-performed model using the same set of hyper-parameters on four datasets. Pol-t, Mol-t, Far-t, and Nap-t stand for Polyvore test set, Modaoperandi test set, Farfetch test set, and Net-A-Porter test set respectively.

Training Set	pairwise AUC				compatibility AUC			
	Pol-t	Mod-t	Far-t	Nap-t	Pol-t	Mod-t	Far-t	Nap-t
Polyvore	**.718**	.685	.594	.636	**.902**	.724	.646	**.749**
Modaoperandi	.589	**.713**	.566	.601	.660	**.732**	.592	.636
Farfetch	.538	.575	**.707**	.620	.560	.585	**.734**	.661
Net-A-Porter	.577	.617	.544	**.668**	.616	.632	.559	.730
Variance	.0061	.0040	.0052	.0008	.0227	.0051	.0058	.0029

Table 5: This table shows the average number of item per outfit in each of the 4 test set and the standard deviation(SD).

Polyvore testset		Moda testset		Farfetch testset		Nap testset	
Mean	SD	Mean	SD	Mean	SD	Mean	SD
5.35	1.62	2.51	.74	3.36	1.15	4.77	1.65

rate is $h(t;\mu,\sigma) = 1 - \Phi((t-\mu)/\sigma)$. The AUC is then $\int_0^1 hdf = \int_{-\infty}^{\infty} h(t;\mu,\sigma)\left[\frac{1}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}\right] dt$. This integral either does not appear in standard tables (or we were unable to find it there; the shift seems to be the problem). But clearly if $\mu_1 > \mu_2$, $h(t;\mu_1,\sigma) > h(t;\mu_2,\sigma)$, so $A(\mu_1,\sigma) > A(\mu_2,\sigma)$. Similarly, if $\sigma_1 < \sigma_2$, $h(t;\mu,\sigma_1) > h(t;\mu,\sigma_2)$, so $A(\mu,\sigma_1) > A(\mu,\sigma_2)$.

But outfit compatibility is computed by averaging the prediction over all edges in the outfit. Considering a dataset where all outfits have G pairs, the compatibility score for a random outfit will still be distributed as $N(0,1)$. The compatibility for a true outfit will be distributed as $N(\mu, \frac{\sigma}{\sqrt{G}})$. In turn, as G grows the compatibility AUC must grow, even if the quality of prediction for each particular pair does not change. This means that compatibility AUC can mislead when comparing methods across datasets. Note that $\sqrt{G}$ is *linear* in the size of outfits, meaning growth could be quite fast. Pairwise AUC (the AUC for all pairs of item under the compatibility score) does not suffer from this effect, and so is better for comparing between datasets.

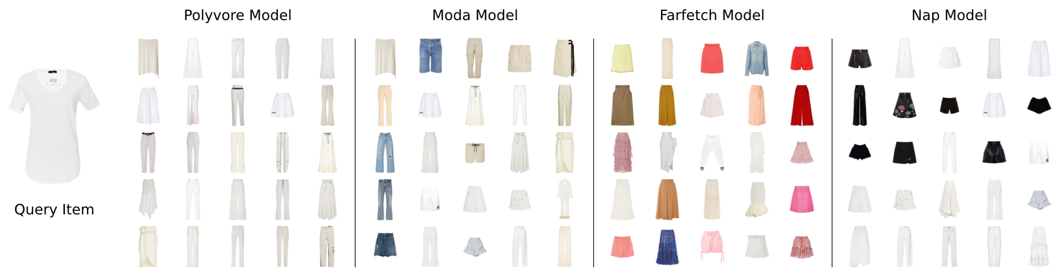

Figure 2: The table shows results from querying the Moda test set using models trained with different datasets (25 items randomly selected from the top 5%). The models *could* return the same set of items in each case, and the difference in model is entirely attributable to the statistics of the training datasets. The figure shows that statistics must vary strongly between training dataset. A model trained on the Polyvore data strongly prefers color matches; one trained on Moda data prefers matches to muted colors; one trained on Farfetch allows quite adventurous color matches; one trained on Nap likes black-white matches.

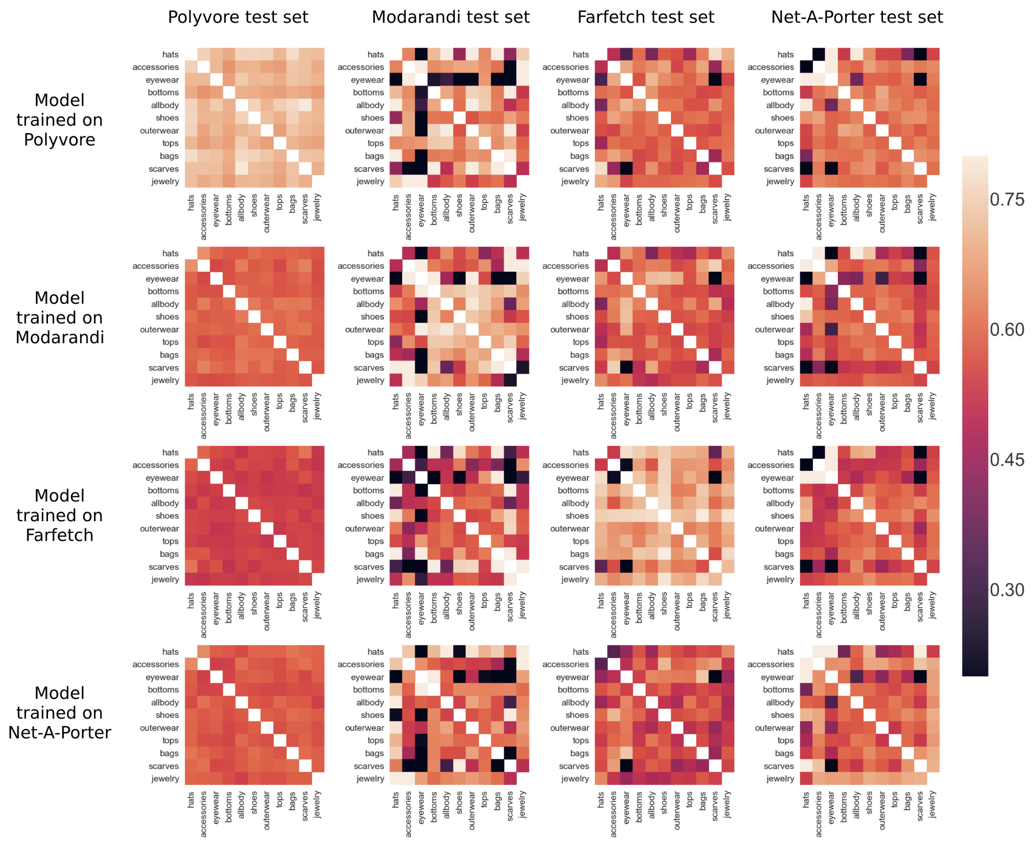

Figure 3: The heat map visualizes the per-type AUC scores of the best-performed model trained using four datasets respectively and tested on the four datasets. It is visibly better to test on the same dataset used for training, but compatibility AUC obscures this fact. As figure 2 shows, different datasets have quite different compatibility properties. Note that for each of the e-commerce dataset, some pairs of types yield poor results.

5.2 Experiment

We use the full transformation function (dot + diff + sum) with 512 dimensions item feature vectors. Other parts of the experiment setup are identical, as described in Section 4.2. We train the model on the four available outfit datasets (3 e-commerce + Polyvore) respectively and test every model on each of the four datasets. For each of the 16 conditions, we compute both the pairwise AUC scores and the compatibility AUC scores.

The results in Table 4 and 5 show that compatibility AUC is consistently biased by the average outfit size of the testing set, which supports the earlier claim. On compatibility AUC scores, we observe the unlikely event that the model trained on Polyvore outperforms the model trained on Net-A-Porter when tested on the Net-A-Porter testing set. This event is an artifact of the outfit size, and the pairwise AUC scores do not reflect such a fact. Notice that pairwise AUC scores of training and testing on the same dataset falls into a smaller range than the compatibility AUC scores. This observation also suggests that pairwise AUC is a more stable metric for cross-dataset comparison of the same method. Further discussion only considers the results from pairwise AUC.

The results show the discriminative method for learning compatibility performs reliably well for different outfit datasets. The scores show that the compatibility link used by professionals are qualitatively different from those used by Polyvore users. Compatibility relationship on each site is also very different: training on one does not get good results when predicting on others. In Figure 2, we query for compatible bottoms to match a plain white t-shirt from the same pool of selectable products. The query results suggest that each dataset has different matching rules for a plain white

Table 6: The table shows the type-pair AUC scores obtained by querying always using the direct strategy vs. combining the direct and indirect approaches. The table also shows the improvement gained by using the combined approach. The AUC scores are averaged over the 11 least frequently occur type-pairs in the training set of each dataset.

Moda testset			Farfetch testset			Nap testset		
direct	combined	gain	direct	combined	gain	direct	combined	gain
.615	.676	.061	.584	.615	.029	.573	.584	.011

t-shirt with some degree of overlap (all match based on color white). We also observe that training on Polyvore consistently gives the second best result when testing on every e-commerce dataset. Both the quantitative and qualitative findings suggest that the compatibility links in Polyvore dataset are more conservative and generalizable than the relationships in the e-commerce datasets.

6 Indirect querying

Some type-pairs yield poor results (as shown in Figure 3), likely because there are few positive training examples for those pairs. Variance effects mean that we will see a poor AUC for those pairs using the **direct strategy** of computing compatibility as a function of items. We show that an indirect computation of compatibility can help.

Our **indirect strategy** works as follows. Assume we wish to compute the compatibility between an item $\mathcal{I}_a$ of type A and an item $\mathcal{I}_b$ of type B. Rather than computing $s(\mathcal{I}_a, \mathcal{I}_b)$, we obtain a set $\mathcal{C}$ of k items of type C, and compute

$$s_{ind}(\mathcal{I}_a, \mathcal{I}_b) = \max_{\mathcal{I}_c \in \mathcal{C}} s(\mathcal{I}_a, \mathcal{I}_c) s(\mathcal{I}_c, \mathcal{I}_b)$$

(i.e., find an item of type C that is compatible with both, and attribute the compatibility from that). The performance depends on the selection of C. At training time, we search for a strategy as follows: for each of the least populous 11 pairs (i.e., 20% of the pairs) of types, compute the AUC for that pair using the direct strategy and the indirect strategy for all possible third types. From the search result, we select the strategy with the best AUC for each type pair, as a **indirect strategy**. This fixed strategy is used at test time for that pair of types.

Indirect strategy results: To evaluate whether there is any advantage in using an indirect strategy, we compare two cases (Table 6) using the models of section 5.2 trained for that experiment. In the first, we use only the direct strategy on test data (direct). In the second, we use whichever strategy emerged from the search at training (combined, $k = 100$). We evaluate by comparing the average of the type pair AUC's, confined to the 11 that could have changed. Using the combined strategy yield a small but useful improvement on test. The search spaces are also sufficiently small that the improvement generalizes.

7 Conclusion

This paper introduced three new, professionally curated fashion outfit dataset for learning fashion compatibility. The work proposed using discriminative method to learn pairwise compatibility relationships and demonstrated that it outperforms the-state-of-the-art on an established dataset, and perform consistently well across different datasets. We proposed pairwise AUC as a new metric for evaluating compatibility tasks across datasets. The work also analyzes the model's performance on different pair of types and demonstrates that an indirect query strategy improves the results for rarely occurred type pairs. Our cross-datasets comparison also indicates that fashion compatibility is a subjective notion - even experts do not have a unified standard. Therefore, we consider explainability and personalization two important future directions.

References

Ziad Al-Halah, Rainer Stiefelhagen, and Kristen Grauman. 2017. Fashion Forward: Forecasting Visual Style in Fashion. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 388–397.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*.

Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. 2017. Learning Fashion Compatibility with Bidirectional LSTMs. *ACM MM 17* abs/1707.05691 (2017).

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ruining He, Charles Packer, and Julian McAuley. 2016. Learning Compatibility Across Categories for Heterogeneous Item Recommendation. *2016 IEEE 16th International Conference on Data Mining (ICDM)* (2016), 937–942.

Wei-Lin Hsiao and Kristen Grauman. 2017. Learning the Latent "Look": Unsupervised Discovery of a Style-Coherent Embedding from Fashion Images. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 4213–4222.

Yang Hu, Xi Yi, and Larry S. Davis. 2015. Collaborative Fashion Recommendation: A Functional Tensor Factorization Approach. In *ACM Multimedia*.

M. Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C. Berg, and Tamara L. Berg. 2015. Where to Buy It: Matching Street Clothing Photos in Online Shops. *2015 IEEE International Conference on Computer Vision (ICCV)* (2015), 3343–3351.

M. Hadi Kiapour, Kota Yamaguchi, Alexander C. Berg, and Tamara L. Berg. 2014. Hipster Wars: Discovering Elements of Fashion Styles. In *ECCV*.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *EMNLP* (2014).

Benjamin Eliot Klein, Guy Lev, Gil Sadeh, and Lior Wolf. 2014. Fisher Vectors Derived from Hybrid Gaussian-Laplacian Mixture Models for Image Annotation. *CoRR* (2014).

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*.

Si Liu, Zheng Jason Song, Meng Wang, Changsheng Xu, Hanqing Lu, and Shuicheng Yan. 2012. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012).

Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015a. Image-based Recommendations on Styles and Substitutes. *SIGIR* (2015).

Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015b. Image-Based Recommendations on Styles and Substitutes. In *SIGIR*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems 28*.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI*.

F. Schroff, D. Kalenichenko, and J. Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Edward Yu-Te Shen, Henry Lieberman, and Francis Lam. 2007. What am I gonna wear?: scenario-oriented recommendation. In *IUI*.

E. Simo-Serra, S. Fidler, F. Moreno-Noguer, and R. Urtasun. 2015. Neuroaesthetics in fashion: Modeling the perception of fashionability. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

E. Simo-Serra and H. Ishikawa. 2016. Fashion Style in 128 Floats: Joint Ranking and Classification Using Weak Data for Feature Extraction. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mariya I. Vasileva, Bryan A. Plummer, Krishna Dusad, Shreya Rajpal, Ranjitha Kumar, and David A. Forsyth. 2018. Learning Type-Aware Embeddings for Fashion Compatibility. *ECCV* (2018).

Andreas Veit*, Balazs Kovacs*, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning Visual Clothing Style with Heterogeneous Dyadic Co-occurrences. In *International Conference on Computer Vision (ICCV)*.

Sirion Vittayakorn, Takayuki Umeda, Kazuhiko Murasaki, Kyoko Sudo, Takayuki Okatani, and Kota Yamaguchi. 2016. Automatic Attribute Discovery with Neural Activations. In *ECCV*.

Wei Yang, Ping Luo, and Liang Lin. 2014. Clothing Co-parsing by Joint Image Segmentation and Labeling. *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), 3182–3189.