Visually-Aware Fashion Recommendation and Personalized Fashion Item Generation

Implementation Track Project (Number of group members - 4)

Abstract

High level of subjectivity in fashion designs and styles, new items continually being introduced, and user preferences and product styles changing over time makes it challenging to build an effective fashion recommender system. Kang et al. introduced a 'fashion aware' recommendation system, called visually-aware deep Bayesian personalized ranking (DVBPR), that incorporates item images and outperforms previous methods, many of which only consider user-item interactions [1]. Furthermore, their paper introduces a generative adversarial network (GAN) to create novel fashion item images and a method to use the GAN and DVBPR together to generate personalized items [1]. Our project implements DVBPR and the GAN. The original paper used a large dataset that included user reviews and vendor-supplied product images [1]. The novelty of our project lies in using a new, smaller dataset that has lower resolution images, and includes customer submitted images alongside the product images from vendors We also use multiple images for each item when training DVBPR whereas the original paper only utilized one image per item. Our DVBPR results indicate that using multiple images per item and incorporating customer submitted images in training can improve performance. On the other hand, utilizing customer and lower resolution images has resulted in worsened performance in the generation of fashion items, however, the generated items are similar to nearest neighbours in the training dataset.

1 Motivation

1.1 Fashion Item Recommendation

With the increase in the number of online shoppers as well as available services, the quantity of visual and textual information on the internet is rising rapidly leading to a pressing need for an intelligent recommendation system which can analyze user's behavior amongst multiple domains and help them find desirable information without the burden of search. Intelligent fashion recommendation system has a great deal of importance in the fields of computer vision and machine learning due to the potential of making huge profits in the fashion industry. Analyzing users preferred styles helps clothing companies understand about the likes and dislikes of the users which can increase their connectivity with the clients. However, there are many challenges to overcome for a fashion recommendation model: a) diversity of fashion styles and availability of large amounts of online information has an impact on retrieving the most accurate options pertaining to a user. b) fashion recommendation models require a large dataset for training (due to seasonality effects) c) user preferences and styles change over time. Due to these challenges, traditional recommendation systems are not sufficient, and fashion-aware recommendation systems are needed.

1.2 Fashion Item Generation

Machine Learning has made huge progress in increasing the human level understanding across the spectrum of perception, reasoning and planning for a machine. A relatively understudied branch is creativity where the goal of the machine is to generate original items with realistic and aesthetic attributes [2]. Fashion is an interesting domain as designing new items requires a lot of creativity coupled with constraints that the items must be wearable. Fashion image generation opens the door

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Table 1: Dataset statistics for a representative dataset from the original paper (Original Amazon Fashion) and the dataset used in this paper (New Amazon Fashion). Image widths vary.

Dataset	Users	Items	5-core Users	5-core Items	Image Types	Image Height
Original Amazon Fashion	64583	234892	45184	166270	Vendor Only	444 pix
New Amazon Fashion	883636	186627	2632	17173	Customer and Vendor	Customer: 88 pix Vendor: 50 pix

for breaking creativity by designing your own clothes. From a limited number of images, a variety of new, fashionable and trending images can be designed with minimal labor and expenses. The new styles may be more appealing than the original designs. Based on GANs, our model takes fashion item images as input and generates a new set of clothes.

2 Problem statement

Given a dataset including user-item interactions and item images, the original paper has three goals. The first goal is to create a visually-aware fashion item recommendation system [1]. In other words, create a system that considers user-item interactions and item images and can accurately assign user-specific preference scores to fashion items a user has not yet interacted with [1]. The second goal is to create a system based on GANs that can be trained on the fashion item images and used to generate novel fashion item images[1]. Finally, the third goal is to combine the recommendation system with the synthesis system to generate novel item images that are personalized to specific users [1].

The goal of our project is to implement the recommendation and synthesis systems, test them on a dataset that is substantially different from those used in the original paper, and evaluate how the results are affected by the new data. In addition, we experimented with a different way of utilizing the dataset when training the recommendation system.

The original paper uses several datasets, but they are all similar to each other and so we will only provide comparisons with one representative dataset here and we will refer to it as the Original Amazon Fashion dataset [1]. The Original Amazon Fashion dataset is a subset of the 2014 Amazon Review dataset [3]. The dataset we will be using is the fashion subset of the 2018 Amazon Review dataset and we will refer to this as the New Amazon Fashion dataset [4]. The New Amazon Fashion dataset was published after the original paper and differs from the Original Amazon Fashion dataset in substantial ways, as outlined in Table 1. While the total number of users is larger in the new dataset, there are fewer items and not every user or item is usable (this will be discussed further in later sections). The 5-core users are users in the dataset that have submitted at least 5 reviews. The 5-core items are items in the dataset that were reviewed by at least one 5-core user. These are the only users and items used to train the recommendation system. Note that the original dataset has a much larger number of 5-core users and 5-core items and much higher resolution images and the new dataset includes customer submitted images and vendor images while the original dataset only has vendor images. Examples of the various types of images are shown in Figure 1. The vendor images are fairly standardized: they are often centered and in front of a white background. The customer submitted images are more variable.

3 Related Work

There is a significant amount of work applying machine learning to fashion that can be broadly categorized into detection, analysis, synthesis, and recommendation [5]. In most works detection methods aim to identify and segment articles of clothing and other items, analysis methods learn to distinguish styles and predict trends, synthesis methods facilitate virtual try-ons, and recommendation methods assess compatibility and match or complete outfits [5]. The recommendation method, DVBPR, that we have implemented is unlike many in this field because it is personalized to specific users [1]. Additionally, the synthesis method we have implemented creates entirely new items, rather than transferring existing items to a new image (e.g. for a virtual try-on) [1]. In general, recommendation systems can be divided into three categories: collaborative filtering, content-based recommenders, and hybrid recommenders [6]. Collaborative filtering methods take into account

Figure 1: Various types of images from the datasets. (Left) a vendor image from the New Amazon Fashion Dataset (Middle) a customer submitted image for the same item in the New Amazon Fashion dataset (Right) a vendor image from the Old Amazon Fashion dataset. Differences in backgrounds and resolutions are apparent.

interactions between users and items (e.g. user movie ratings) but do not consider qualities of the items themselves (e.g. movie genre or visual content). Content based recommenders focus on comparisons across items' and users' auxiliary information [6], such as the text content of news articles and topics of interest selected by users. Hybrid recommenders integrate both types of recommendation strategies [6]. Some recommenders may additionally account for temporal and social information [7]. DVBPR is a hybrid recommender that considers user feedback and item images. The method's visual-awareness makes it much more effective than those that consider user/item interactions alone, but it does not account for temporal or social information. Matrix factorization (MF) is a successful collaborative filtering method that maps both users and items to a so-called latent factor space where the dot product of user and an item in this space indicates the interest the user has in the item [8][9]. DVBPR builds upon MF by determining a visual latent space representation of the users and items based on user/item interactions and item images [1]. The same authors previously proposed a similar method, VBPR, that uses a pre-existing CNN, trained for a more general image classification task, to extract image features and ultimately map item images to the visual latent space [10]. DVBPR improves upon VBPR by training a CNN directly to map item images to the visual latent space [1]. DVBPR uses a generic optimization criterion for personalized ranking called BPR as the objective function used in training [1][11].

4 Methodologies

This section describes the frameworks and implementations of the methods. We found that a complete re-implementation of these methods was too much work under the given time constraints and we leveraged existing code provided by the original paper's authors [12]. However, in order to make the existing code robust to low resolution images, we had to write code to preprocess the new dataset, as well as modify the existing code for DVBPR and the GAN in terms of architecture and tune hyper-parameters as described in this section. We also ran our own experiments on the new dataset.

4.1 DVBPR (Fashion Item Recommendation)

DVBPR Framework Let U be the set of users, I be the set of items, I_u^+ be the subset of items about which the user $u \in U$ has expressed positive feedback, and I_u^- be the subset of items with no feedback from user u. The goal of DVBPR is to be able to assign a preference score to each item in I_u^- for a user u as follows [1]:

$$x_{u,i} = \theta_u^T \Phi(X_i) \tag{1}$$

where X_i is an image associated with an item $i \in I_u^-$, θ_u is a latent visual preference vector for user u and Φ is a CNN network trained to map product images to the same latent space [1].

The CNN architecture used is based on one designed for efficient training called CNN-F [13]. The specific architecture is given in Figure 2 [1]. The input image size of CNN-F is 224x224 and therefore all images are resized to these dimensions during training [13].

The objective function used to train the model is called Bayesian Personalized Ranking (BPR) [11]. BPR is a general optimization criterion used for personalized ranking that has been found to be

conv1	conv2	conv3	conv4	conv5	full6	full7	full8
64x11x11	256x5x5	256x3x3	256x3x3	256x3x3	4096	4096	K
st 4, pad 0	st 1, pad 2	st 1, pad 1	st 1, pad 1	st 1, pad 1	Drop out	Drop out	-
x2 pool	x2 pool	-	-	x2 pool	-	-	-

Figure 2: The CNN architecture used in DVBPR (st=stride, pad=spatial padding).

successful [11]. The intuition behind BPR is that the system should assign a larger preference score to items the user has expressed positive feedback about than it does to items the user has expressed no feedback about. Formally, define the set $\mathcal{D}$ as [1]:

$$\mathcal{D} = \{(u, i, j)|u \in U \wedge i \in I_u^+ \wedge j \in I_u^-\} \tag{2}$$

This is a set of triplets (u, i, j) where u is a user, i is an item they have expressed positive feedback about, and j is an item they have not. The difference in preference scores between i and j is [1]:

$$x_{u,i,j} = x_{u,i} - x_{u,j} \tag{3}$$

The BPR objective function is then given as [1]:

$$max \sum_{(u,i,j)\in\mathcal{D}} \ln \sigma(x_{u,i,j}) - \lambda_\Theta ||\Theta||^2 \tag{4}$$

where σ is the sigmoid function, Θ includes all the model parameters, and λ_Θ is a regularization hyperparameter. Note that evaluating $x_{u,i,j}$ requires using two CNNs: one on the image X_i and one on the image X_j. In DVBPR these CNNs are made to share the same weights, a method that is called a Siamese network [1][14]. The optimization is performed with the Adam optimizer, a stochastic gradient ascent algorithm [15]. At each iteration a single triplet (u, i, j) is sampled.

The framework is generalizable to various expressions of positive feedback. In the original paper and in our project any review of an item, regardless of whether it is a positive review, is taken as an expression of positive feedback between the reviewer and the item [1].

Data Preprocessing We preprocessed the New Amazon Fashion dataset in three different ways to evaluate which method of using the data is best. In the first method we used vendor images only and only one image per item. This matches what was done in the original paper [1]. In the second method we also used vendor images only, but we used every vendor image available. Conceptually, this amounts to representing a single review for a particular item as multiple reviews, one for each image of that item, therefore allowing the set I_u^+ to be expanded without requiring more reviews. In the third method we used all vendor images and all customer submitted images, which allowed the set I_u^+ to be expanded even more, but potentially introduced low quality images. We will refer to the resulting preprocessed datasets as Vendor Only Single Image, Vendor Only, and Vendor Plus Customer.

In all of the preprocessing methods the dataset was first reduced to only reviews given by 5-core users and items reviewed by 5-core users (the 5-core items). Then the images were compiled for each 5-core item (according to the methods described above). Some 5-core items included no images or their URLs to item images were broken and such items were removed. Some duplicate reviews also had to be removed. If a user did not have five reviews for items with images after these removals, the user was removed entirely. The preprocessed dataset statistics for the three methods used are listed in Table 2 alongside the statistics from the Original Amazon Fashion dataset that was used in the original paper.

4.2 GAN (Fashion Item Generation)

DCGAN Framework Guiding the shoppers and designers into exploring the field of potential fashion designs and styles (which are currently not available in market) is a richer form of recommendation. For the GANs framework, an architecture called Deep Convolutional Generative Adversarial

Table 2: Preprocessed dataset statistics for the three methods of preprocessing the New Amazon Fashion dataset and for the preprocessed dataset used in the original paper.

Preprocessed Dataset	Image Usage	5-core Users	5-core Item Images
Vendor Only Single Image	1 vendor image per item	2326	8579
Vendor Only	All vendor images	2323	33587
Vendor Plus Customer	All vendor and customer images	2377	38053
Original Amazon Fashion	1 vendor image per item	45184	166270

Networks (DCGAN) was used to generate fashion images given a product category which is most consistent with it [16]. GANs are used to transform the discrete space fashion images to a continuous space by generating a wide range of fashion images for each product category. GANs is an unsupervised learning framework which comprises a generator and a discriminator which are typically multilayer convolutional and de-convolutional neural networks. The generator and discriminator compete against each other to generate realistic looking images, which cannot be distinguished from the training corpus. This is achieved by training the generator to generate images similar to prior distribution and training the discriminator to classify if the image is real or synthetic. The generator and discriminator in DCGAN are trained simultaneously to ensure that the synthesized images have the same distribution as the images from the training data, so that synthesized images are as close to training images as possible. The generator $G(z, c)$ takes a random noise vector as an input, $z \sim U(-1, 1)$ and a category c to synthesize an image. The discriminator inputs an image x sampled from the training corpus X_c or a generator-synthesized image $G(z, c)$ and determines if the image is real or fake. Despite the architecture being based on DCGAN, least squares loss is used (which is used in LSGAN) to train the GANs framework. Least squares loss avoids loss saturation leading to a stable training process. The result is that the generated images are of better quality and larger size than in standard DCGAN [17]. The objective functions of the generator and discriminator are [1]:

$$\min_{\mathbf{G}} V(G) = \mathbb{E}_{c\sim p(c), z\sim p(z)} L_{real}(G(z, c), c) \tag{5}$$

$$\min_{\mathbf{D}} V(D) = \mathbb{E}_{x,c\sim p_{data}(x,c)} L_{real}(x, c) + \mathbb{E}_{c\sim p(c), z\sim p(z)} L_{fake}(G(z, c), c) \tag{6}$$

where $L_{real}(x, c) = [D(x, c) - 1]^2$ and $L_{fake}(x, c) = [D(x, c)]^2$. Thus, the discriminator predicts 1 for real images and 0 for fake images. The model is optimized by using mean squared error (MSE) loss function (L2 loss). The output layer of the discriminator is a linear activation function.

Implementation and Architecture Details for DCGAN The GANs framework is conditioned on a product's top level category. The objectives for generator and discriminator are optimized alternately by keeping the epochs = 55 until the quality of generated images obtained is enough to distinguish between the fashion items. As the dataset we were using had lower resolution images, the overall quality of images in the training dataset were not as good as the images from the dataset used in original paper. As a result we had to make changes in the details of architecture for the generator and the discriminator. The overall skeleton and the number of layers for the generator and the discriminator were kept the same as the paper, however we changed the filter, stride and input-output layer size of the architecture. This enabled decent quality images at the output despite the low resolution of the images. Figure 3 shows the generator and discriminator architecture that we implemented in our project in detail. Except for the last layer of the generator where it uses tanh, the architecture for generator and discriminator uses leaky relu as the activation function. The Leaky relu activation function helps in non-saturation of the gradient and hence accelerates the convergence.

Data Preprocessing The New Amazon Fashion dataset contains images belonging to a variety of categories like shoes, sandals, shirts, jeans, skirts, tops, etc.. In this project we generated images for women's tops, pants, and shoes and men's shirts, pants, and shoes and used both customer submitted images and vendor images in training. Upon inspecting the data, we noticed that some of the images were irrelevant to the product category and had to be discarded. We also found that using the full set of images across the desired categories was prohibitively time and memory consuming during training. Therefore, the final size of the image dataset was 10,000. As a final pre-processing step, we had to center crop each image and resize it to 64 x 64.

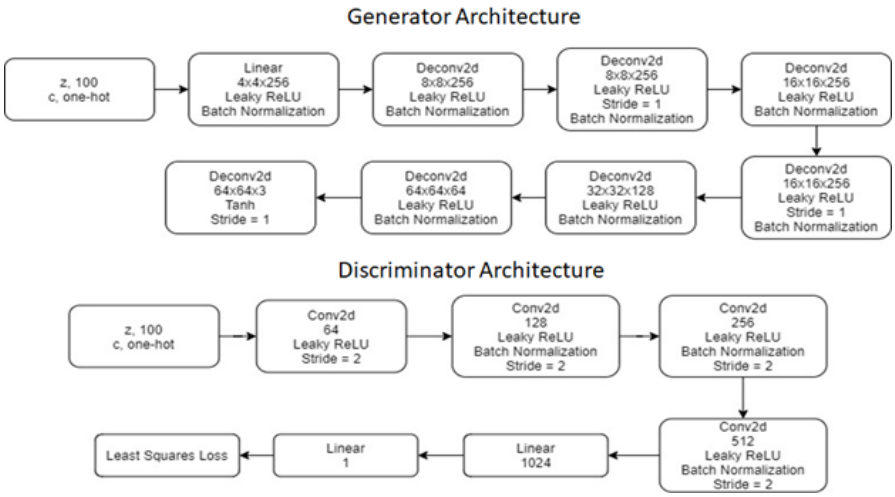

Figure 3: Generator and Discriminator Architecture Details

5 Evaluation

5.1 DVBPR (Fashion Item Recommendation)

As was done in the original paper, we held off one review per user for a validation set and one review per user for a test set and the rest were left for training [1]. The results shown are from the validation set. We use the area under the receiver operating characteristic curve (AUC) as the performance metric for the optimization system. This is given as [1]:

$$AUC = \frac{1}{|U|} \sum_{u \in U} \frac{1}{|\mathcal{D}_u|} \sum_{(i,j) \in \mathcal{D}_u} \mathcal{E}(x_{u,i} > x_{u,j}) \tag{7}$$

where $\mathcal{E}$ is an indicator function. This metric essentially counts the number of times items that have received positive feedback are given a higher preference score than items that have received no feedback [1]. The experiments were conducted on a single machine with a NVIDIA 1080 Ti GPU, Intel Core i7-7700K CPU, and 32 GB of RAM. Each run took approximately three hours to complete. Due to the duration of the runs, and because the original paper states that DVBPR is not sensitive to hyperparameter tuning, we used the same hyperparameters as the original paper: a mini-batch size of 128 in training; a latent factor dimension of 50, weight decay of 10^{-3} and probability of dropout set to 0.5 in the CNN; and lambda set to 1 in the BPR objective function [1]. The training was done for each preprocessed dataset for 14 epochs.

The AUC results yielded from training on the three preprocessed datasets of the New Amazon Dataset are given in Table 3 alongside the results given in the original paper that were yielded from training on the Original Amazon Dataset [1]. The performance from the original paper using the original dataset is superior, but this is not surprising because that dataset included many more 5-core users and items and item images with a higher resolution. The results on the other datasets are more interesting. These results indicate that performance improves as more images are utilized going from single vendor images, to all vendor images, to all vendor and customer submitted images. This is an interesting discovery because the original paper only used one image per item and this implies that they may have been able to do more with the same dataset had they used multiple vendor images. It is also interesting to see that including customer images improves recommendation performance, despite the customer images being far less standardized than the vendor images.

5.2 GAN (Fashion Item Generation)

For training we had to tweak a few parameters from the original implementation in order to enhance the quality of images of fashion items that were generated.

Table 3: DVBPR performance on each preprocessed dataset in terms of AUC.

Preprocessed Dataset	**AUC (Higher is Better)**
Vendor Only Single Image	0.6501
Vendor Only	0.6853
Vendor Plus Customer	0.6875
Original Amazon Fashion	0.7964

All input images were resized to 64 X 64 X 3. The computations were conducted on a single machine with a NVIDIA 1080 Ti GPU, AMD Ryzen 1700X CPU, and 32 GB of RAM. The total run-time for the optimized hyperparameters on this machine was 7h 06m 36s. For hyperparameters, the learning rate was tuned to 1e-5 and batch normalization [18] was applied for all convolutional and deconvolutional layers except for the first and the last layers, to make the weights in deeper layers robust to changes in the earlier layers and for ease of training. It also helped to deal with poor initialization and in the gradient flow. The training was done for 55 epochs, which was significantly longer than the original paper which only used 30 epochs, because the dataset we were using had a combination of customer submitted images and vendor images and both were of lower resolution. The Adam optimizer was used for optimization [15]. The Adam optimizer combines the RMS Prop and Gradient Descent with momentum techniques to make it more robust. We used the Adam Optimizer with $\beta_1 = 0.9$ (First momentum) and $\beta_2 = 0.999$ (Second Momentum). The weight decay regularization coefficient was kept the same as the original paper at 1e-5 [1]. The minibatch size was changed from the original paper's value of 16 to 64. Changing the minibatch size reduced the training time as well as improved image quality for the test data. The dataset was also shuffled before training in order to even out the distribution of the data. Qualitative assessment of the images showed that the generated clothing items were more diverse than those from the training corpus. The images generated weren't exactly identical to those in the training dataset and were also not too noisy. Figure 4 shows sampled images generated by our GAN framework from each of six categories: women's tops, pants, and shoes and men's shirts, pants, and shoes. These demonstrate that the generated images are realistic and plausible, yet are different from images in the original dataset in the sense that they have common shape and color profiles but quite different styles.

Figure 4: Images generated from GANs

For a given product category, a lot of images in the women's category have pink coloured tops and pink coloured shoes, as seen in Figure 4. After inspecting the training images, it was evident that a

lot of women customers purchased pink coloured tops and shoes and hence pink shoes and tops were generated by the GANs framework more predominantly in women's category than men's category. The shoes generated in the men's category had more colors inclined towards black, gray and brown and were mostly laced with no heels or stilettos, which is also in alignment with the training images. In order to further evaluate the GANs framework, we also found the nearest neighbours in the training dataset to the images generated by the GANs. The L1-approximated images were found to be similar to the given prototype and most images were in vicinity of the original image. While the images generated are different than those in the training corpus, they have similar style indicating that the item's individual styles have been effectively captured.

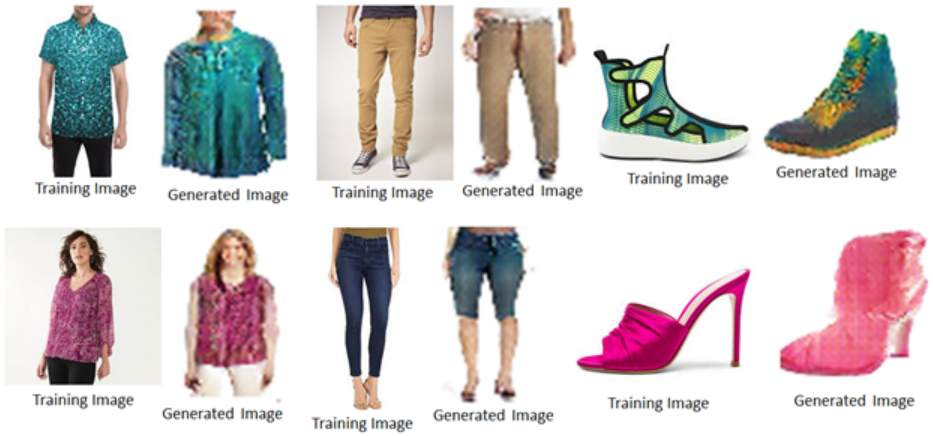

Figure 5: Vendor Selected Product Images (left) vs Generated Product Images (right)

Figure 5 highlights the continuous nature of the space learned by GAN. The types of modifications vary for different users such as changing the color, extending sleeve length, shortening of pants, changing from open-toed to closed-toed heels, or changing orientations and other minor stylistic changes. This evaluation showed that generative model potential can uncover desirable items that did not exist in the dataset and generate novel fashionable designs.

6 Conclusion

We implemented two fashion based systems from a paper: one that can be used for recommendation as well as one that can be used for design [1]. We trained and tested both systems with a dataset substantially different from those used in the original paper [4]. In particular, the new dataset had lower resolution images and included customer submitted images in addition to the images from vendors. The recommendation system, DVBPR, assigns items preference scores based on user feedback and item images [1]. In general, DVBPR's performance on the new dataset was worse, likely due to the lower resolution and smaller amount of usable data. However, we discovered two new ways of using data for DVBPR, using multiple images per item and using customer submitted images, that worked better for the new dataset and might be better in general. The framework of GANs learns the distribution of fashion images and generates novel items [1]. Due to the use of customer submitted images in the dataset and the lower image resolution, the quality of the images is not up to par with the original work's dataset and hence we learned that the performance of our model wasn't as good as the paper had presented. Due to use of low quality images, the generation of fashion items was also seen to have a lot of noise while generating tops and shirts for men and the designs and prints on the items weren't seen to be consistent. However, we were able to generate fashion items that were novel in nature, such that they did not match the training images but at the same time confirmed the shape, size and colour of their nearest neighbours in the training dataset. A possible future direction to this work will be improving the quality of the generated images and providing control of fine-grained styles. Apart from that, DVBPR-GANs framework can also be extended to applications besides fashion images to provide richer recommendations like household items, jewelry, and others.

References

[1] W.-C. Kang, C. Fang, Z. Wang, and J. McAuley, "Visually-aware fashion recommendation and design with generative image models," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 207–216.

[2] S. Jaradat, "Deep cross-domain fashion recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 407–410.

[3] J. McAuley, "Amazon product data," 2014. [Online]. Available: http://jmcauley.ucsd.edu/data/amazon/links.html

[4] J. Ni, "Amazon review data (2018)," 2018. [Online]. Available: https://nijianmo.github.io/amazon/index.html

[5] W.-H. Cheng, S. Song, C.-Y. Chen, S. C. Hidayati, and J. Liu, "Fashion meets computer vision: A survey," *arXiv preprint arXiv:2003.13988*, 2020.

[6] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–38, 2019.

[7] R. He, C. Fang, Z. Wang, and J. McAuley, "Vista: A visually, socially, and temporally-aware model for artistic recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016, pp. 309–316.

[8] S. Funk, "Netflix update: Try this at home," Dec 2006. [Online]. Available: http://sifter.org/simon/journal/20061211.html

[9] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender systems handbook*. Springer, 2015, pp. 77–118.

[10] R. He and J. McAuley, "Vbpr: Visual bayesian personalized ranking from implicit feedback," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI'16. AAAI Press, 2016, p. 144–150.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, ser. UAI '09. Arlington, Virginia, USA: AUAI Press, 2009, p. 452–461.

[12] W.-C. Kang, C. Fang, Z. Wang, and J. McAuley, "Dvbpr," 2017. [Online]. Available: https://github.com/kang205/DVBPR

[13] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[14] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[15] D. Kingma and J. Ba, "Adam: A method for stochastic optimization, 3rd international conference on learning representations,{ICLR}," 2015.

[16] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2016.

[17] J. Brownlee, "How to develop a least squares generative adversarial network (lsgan) in keras," Sep 2020. [Online]. Available: https://machinelearningmastery.com/least-squares-generative-adversarial-network

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.