Semi-supervised Adversarial Learning for Complementary Item Recommendation

Koby Bibas
kobybibas@gmail.com
Meta
Tel Aviv, Israel

Oren Sar Shalom*
oren.sarshalom@gmail.com
Amazon
Tel Aviv, Israel

Dietmar Jannach
dietmar.jannach@aau.at
University of Klagenfurt
Klagenfurt, Austria

ABSTRACT

Complementary item recommendations are a ubiquitous feature of modern e-commerce sites. Such recommendations are highly effective when they are based on collaborative signals like co-purchase statistics. In certain online marketplaces, however, e.g., on online auction sites, constantly new items are added to the catalog. In such cases, complementary item recommendations are often based on item side-information due to a lack of interaction data. In this work, we propose a novel approach that can leverage both item side-information and labeled complementary item pairs to generate effective complementary recommendations for cold items, i.e., for items for which no co-purchase statistics yet exist. Given that complementary items typically have to be of a different category than the seed item, we technically maintain a latent space for each item category. Simultaneously, we learn to project distributed item representations into these category spaces to determine suitable recommendations. The main learning process in our architecture utilizes labeled pairs of complementary items. In addition, we adopt ideas from Cycle Generative Adversarial Networks (CycleGAN) to leverage available item information even in case no labeled data exists for a given item and category. Experiments on three e-commerce datasets show that our method is highly effective.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Recommender systems, Complementary Items, CycleGAN

ACM Reference Format:
Koby Bibas, Oren Sar Shalom, and Dietmar Jannach. 2023. Semi-supervised Adversarial Learning for Complementary Item Recommendation. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3543507.3583462

*Work was done while with Meta.

The Web Conference '23, April 30–May 4, 2023, Austin, TX

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583462

1 INTRODUCTION

Many modern e-commerce sites provide *complementary* item recommendations for their customers. For example, when online users shop for shirts, they may receive suggestions for suitable pairs of pants. In practice, such recommendations often appear under labels such as "Frequently bought together", and these recommendations are an important driver for cross-selling on e-commerce platforms [5, 17, 18]. Given the typically large product assortments in e-commerce, the ranking of complementary item recommendations is commonly based on behavioral patterns observed in the consumer community, for instance on co-purchase patterns. In addition, domain knowledge may be incorporated, for example, that the complementary item recommendations have to come from a pre-defined *target* category (e.g., pants), or from a category that is different from the category of the *seed* item, i.e., the item for which complementary suggestions are sought for.

In some domains, however, item catalogs can be very volatile [30]. On the eBay (*ebay.com*) auction platform, for example, constantly new items become available and often there is only one exemplar of a certain product available [8, 39]. Consequently, suitable methods are needed for *cold* seed items, i.e., items for which no transaction data is available. In this setting it is impossible to model the seed item based on collaborative signals like item co-purchases. Thus, one may have to rely on alternative approaches, like processing side-information about the items, e.g., natural language descriptions of the items or structured item attributes.

Although we focus on cold items, it is still possible to leverage aggregate usage data. Specifically, such data allows us to infer a set of *complementary categories*, i.e., categories that include complementary items, like the categories pants and shirts. Recommendations for an item from a certain category can then be constrained to items that belong to complementary categories. The advantages of this approach are threefold: (1) it helps rule out irrelevant recommendations; (2) it significantly reduces inference time; and (3) it can be used to design bundles [3, 50] (e.g., by recommending a single item per complementary category), and thereby allow users to understand how they were formed. Given these advantages, category-level constraints on complementary item recommendation is therefore a common practice at large e-commerce sites like Facebook Shops and Instagram Shops[1].

In this work, we propose a novel deep learning model to address this challenging problem setting. The main technical idea of our approach is to maintain a latent space for each item category. Simultaneously, we learn to translate representations of the items,

[1] https://research.facebook.com/blog/2021/12/shops-on-facebook-and-instagram-understanding-relationships-between-products-to-improve-buyer-and-seller-experience/

which we derive with the help of side-information, into these latent spaces. This process finally allows us to locate items in a given category space that are close to the seed item. The basic learning process in our architecture is based on *supervised learning* using labeled pairs of complementary items. We note that these pairs of items, which can for example be obtained by analyzing existing shopping baskets of a site, do *not* include the seed item, as we supply recommendations for cold items. Instead, they help us learn how categories are related. Since these labeled pairs can be sparse for certain categories, we furthermore incorporate ideas from Cycle Generative Adversarial Networks (CycleGAN) [49] in our architecture. This allows us to leverage available item information even when no labeled data exists for a given item and category.

Overall, we therefore suggest a novel *semi-supervised* model for complementary item recommendations. An evaluation of the model on three e-commerce datasets confirms that it is favorable over alternative approaches under different experimental settings. Moreover, the experiments clearly demonstrate the benefits of extending the architecture with CycleGAN elements. To our knowledge, utilizing CycleGAN concepts for improved complementary item recommendation has not been explored in the literature before.

2 BACKGROUND AND RELATED WORK

A common technical approach in real-world e-commerce is to determine related items, e.g., on item detail pages[2], is to rely on co-purchasing patterns. Recommendations based on such simple shopping basket analyses can be surprisingly effective [22, 25]. We note, however, that the items returned by such approaches are not necessarily complementary items, and it may happen that two particular shirts are frequently bought together [11]. In our work, we therefore assume that a complementary item belongs to one among the pre-defined complementary categories, which are different from the seed item's category (e.g., pants in the case of shirts).

Another type of recommendations commonly seen in practice are often shown under labels such as "Similar Items" or "Related Items", some of which may serve as substitutes [5, 28, 35, 37, 44]. Finding similar items is another relevant problem, but almost the opposite of our research focus. *Related items*, see, e.g., [37], on the other hand, can in theory include both complements (e.g., accessories) and substitutes, i.e., alternatives. Several works [1, 2, 26] aim to find complementary products, but these often cannot distinguish between similar and complementary items, since the seed and the recommended items are encoded by the exact same network.

In terms of the addressed research challenge, the work by Galron et al. [8] at eBay is closest to our research, and we also adopt a similar evaluation approach. Given the sparsity of user-item interaction data, which hampers the use of traditional collaborative filtering approaches, the item-based DCF (Deep Collaborative Filtering) method proposed in [8] learns a similarity function between items based on their side-information. This similarity function is then used to retrieve recommendations for a given seed item, i.e., the neural network takes a pair of seed and target items as inputs, and returns a similarity prediction as an output. Internally, the network encodes the items as sparse feature vectors based on characteristics such as the title or category. To train the model, purchase data from eBay in the form of pairs of items that were co-purchased by the same users is used. Computational experiments and an A/B test indicate that DCF performed significantly better than the existing system at eBay. Given the proven performance of DCF in a real-world setting, we use this method as a baseline in our research. We note that in the evaluation of DCF recommendations from the same category were considered as ground truth.

P-Companion [11] is another neural framework for complementary item recommendation. This model focuses on diversifying the recommendations and as it outperformed several other methods [10, 27, 40], we also use it as baseline. While this model has some commonalities with our approach, it does not allow semi-supervised learning. Furthermore, it does not maintain a separate latent space per category. Instead, the various dimensions in the item embeddings are weighted differently, according to their category.

In terms of applications, various previous works focus on fashion recommendation [7, 23, 41, 47]. In many cases, *visual* approaches are highly effective [6]. Such approaches typically consider domain-specifics and strongly rely on item images to establish relationships between different clothing items, e.g., by projecting items in a shared visual style space [27]. In a recent work [7], the authors for example use shape and color information to learn which shapes and colors are compatible. Compatibility estimation is also the focus in [14, 34]. A session-based recommendation approach, which also considers visual information is presented in [41]. However, such visual approaches can also have their limitations. Specifically, purely visual approaches may return items that are stylistically similar, but not complementary. In [47], the authors therefore propose a complementary item recommendation approach that relies on textual information, and can be extended using recent methods [32]. These works are different from our approach in that our framework is not domain-specific. Furthermore, our approach considers both visual and textual side-information and in addition allows us to specify target categories for the complementary items. Explicitly making compatibility predictions between items is not in the focus of our present work, but an interesting area for future investigations. Similarly, works that aim to automatically discern if two items rather represent alternatives or complements, e.g., [24, 40], may be integrated in our framework in the future as well to assess if they help further improve accuracy.

A number of other works exist that use datasets that contain information about co-purchased or co-viewed items like we do in our computational experiments, e.g., [36, 48]. The focus of such works however is often not primarily on making complementary item recommendations. Instead, the goal is to improve prediction accuracy in general, which is achieved by exploiting co-occurrence information in the data. No distinction is however made if the resulting recommendations actually contain substitutes or complements.

The use of adversarial training in recommender systems is widespread [19, 20, 29]. Specifically, using GANs for complementary recommendations was proposed in [21], where a generator learns how to create the image of a complementary item. However, the goal of this methods is to generate an image and not an item. Another generative adversarial learning approach is proposed in [16]. Given a seed item, a generator network creates recommendations

[2]Such recommendations are often not personalized according to long-term user profiles, which is also one assumption of our present work. See [39] for a personalized approach at eBay.

and a discriminator needs to distinguish between real labeled recommendations and generated ones. However, this model requires full supervision and, unlike our approach, it does not allow to control important traits of the recommendations, like the category.

A *quality-aware* method for complementary item recommendation is proposed in [46]. The underlying intuition is that not only the compatibility of the items matters, but also the quality of the recommended items. The quality in their approach is based on explicit item ratings, and the proposed model jointly considers user preferences and compatibility aspects. In our work, in contrast, we do not assume the existence of long-term user preference profiles and explicit ratings. Nonetheless, personalizing the recommendations [38, 43, 45] or considering the sequential nature of user-sessions [42] may be an aspect to explore in the future.

3 OVERVIEW OF PROPOSED APPROACH

Summary of Problem Setting. We recall the specifics of our problem setting. Given a *seed* item and a *target* category, the task is to recommend complementary items from the target category[3]. The main challenge is that no interaction data yet exists for the seed item. However, we assume that each item has a known category and there is some additional side-information available for each item. Moreover, we make the assumption that there exists a labeled set of pairs of complementary items[4]. Such a set may be created manually or automatically derived, e.g., by analyzing co-purchase patterns of pairs of items with sufficient purchase signals. Generally, however, paucity of labeled data is an inherent problem, and a model that can cope with it is preferable.

Technical Approach. Let C denote the set of all item categories in the catalog. As mentioned, one key idea of our approach is to maintain a latent space for each item category $c \in C$. To make a recommendation for a given seed item, we first create a distributed representation (i.e., embedding) of it, and we then translate this embedding into the target category's latent space. The translation process can be understood as creating a pseudo-item by converting the seed item's representation into the latent space of the target category. The main challenge in our approach now is to learn based on sparse data how to translate items to the target latent space in a way that the important traits of the seed item are maintained. Once the seed item is positioned in the target category's latent space, plausible recommendations can be derived by selecting items that are close to it. More formally, let I_c be the set of items in category c. To find plausible recommendations in category c for seed item s, the translated representation v_s^c is generated. Then items are recommended according to their distance from v_s^c: $\text{argmax}_{i \in I_c} \cos(v_s^c, v_i)$, where cos is the cosine similarity and v_i is a distributed representation of item i.

We recall that in our approach we aim to concentrate on a subset of *relevant* categories for each item. Therefore, given an item's category, we determine complementary categories from item-level labeled data. To find the set of complementary categories for category $c \in C$ we aggregate the item-level labeled pairs to the category-level to determine the set of other relevant categories[5]

Architecture Components. The overall architecture has two main elements. First, the architecture includes a *supervised* learning component, which uses item side-information and the labeled set of pairs of complementary items to learn the translation between the category latent spaces. Since this labeled set of pairs may suffer from data paucity in particular for rare categories, we extend the architecture to include a sub-network that supports *unsupervised* adversarial learning from all items in the catalog. I.e., also for items for which there is no labeled complementary item in a given category in the dataset. Thus, the two elements combined together constitute a semi-supervised learning approach. Figure 1 shows the overall architecture of the model.

4 SUPERVISED LEARNING COMPONENT

The supervised learning component has two elements. The *item encoder* creates an embedding of the items. The *category translator* then translates the encoded item into the latent space of any given target category.

Item encoder. Given an item i, the model first generates its representation v_i based on its side-information. In many application scenarios, including fashion as discussed above, the image of the item is a highly important piece of side-information. To incorporate this information, we first feed the item image through a pre-trained image processing model. Then, we pass its output through a learned multilayer perceptron (MLP). For categorical features, the model fits an embedding per each distinct value. We note that continuous features, such as an item's price, can be dealt with by converting them to categorical features through discretization. To obtain the overall item representation, the representations of all features are concatenated and passed through another MLP. We acknowledge that other, more sophisticated approaches could be used instead of a simple MLP. Yet, we defer these potential improvements to future work. We note that the category itself is used here as another feature, which makes the *item encoder* category-aware.

Category translator. The task of this element is to learn how to transfer the item representations from one latent space to another. For example, let s be a specific shirt and p and n (stand for positive and negative) are two pairs of pants, with embeddings v_s, v_p and v_n, respectively. Let us assume that s is complemented by p, but not by n. Then, we would like to learn a transformation such that projecting the shirt's representation v_s into the pants domain will yield a representation that is more similar to v_p than to v_n. We stress that there are many plausible architectures to combine the item and the category representations, some of them allow heterogeneous dimensionalities of the latent spaces of the various categories. However, our model of choice is a simple concatenation of these embeddings, followed by an MLP.

Model training. Training in the supervised learning component is based on the given set of labeled pairs of items. Each labeled pair consist of a seed item s and a positive complementary item p of

[3] In practice, there can be multiple complementary categories and the recommendation process may thus be repeated for each category.

[4] Note that the pairs in the labeled set do not contain the seed item.

[5] In our work we do not allow complementary item recommendations from the same category, as done in [41]. This constraint can however be trivially dismissed as needed.

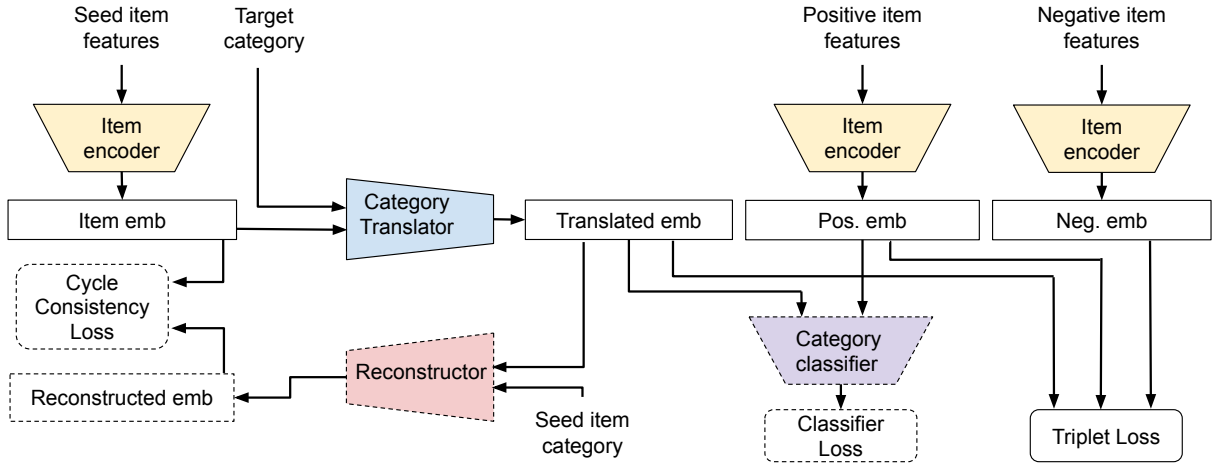

Figure 1: The proposed model architecture. Elements with dashed lines are part of the adversarial learning network.

category c. For each such pair, we apply negative sampling and draw a negative item n (from category c as well). The *item encoder* is then invoked on each of the items to obtain the representations v_s, v_p and v_n. Then, applying the *category translator* on v_s yields v_s^c, the representation of the seed item in the desired latent space. As a loss function, we use the triplet loss [31]: $\mathcal{L}\left(v_s^c, v_p, v_n\right) = \max\left(\mathrm{d}\left(v_s^c, v_p\right) - \mathrm{d}\left(v_s^c, v_n\right) + \alpha, 0\right)$, where d is a distance function; the negative cosine similarity performed well in our experiments. α is a hyperparameter that sets the margin by which the positive sample should be more similar than the negative one.

5 ADVERSARIAL LEARNING COMPONENT

The effectiveness of any *supervised* learning approach is bound by the available labeled data. In complementary item recommendation, data paucity, as mentioned, is a common problem. In particular for more rare categories there may not be a sufficient amount of labeled pairs available for effective learning. Therefore, our architecture includes an adversarial learning component which implements *unsupervised* learning. Thereby, we aim to leverage information from any item in the catalog, even if there are no labeled complementary items for it in a given category. In our example, this would be the case if we are given a particular shirt and there is no complementary item in the pants category in the training data. Ultimately, the combination of both components leads to a *semi-supervised* learning approach. We emphasize that the adversarial component is only needed to further improve the training process. At inference time, only the outputs of the *item encoder* and the *category translator* are used to determine the recommendations.

5.1 Architecture Elements

The main goal of this component is to further improve the effectiveness of the *category translator*. To ensure that the *category translator* is not misguided when incorporating information about items without labels, the adversarial component takes inspiration from CycleGAN [49]. CycleGAN was designed in the context image-to-image translation problems, e.g., to translate a summer landscape into winter. A central idea in these networks is the concept of *cycle consistency*. For the mentioned example, a translation would be cycle consistent if we end up close to the original image if we translate the winter landscape back. Ultimately, cycle consistency ensures that inputs and outputs are paired in a meaningful way.

Technically, our adversarial learning component has two main sub-networks, the *classifier* and the *reconstructor*. The *classifier* receives an item representation v_i and a category c as an input and returns the probability that item i belongs to category c. To this end, the *classifier* assigns a score for each category by passing v_i to an MLP, where the last layer is of size $|C|$. Then, these scores are turned into probabilities using *softmax* and the probability of the item belonging to category c is extracted. Finally, the translated representation v_i^c is sent to the *classifier* to affirm it indeed looks like a representation of an item in category c. This auxiliary component motivates the *category translator* to produce reliable embeddings. However, it might be incapable to train a good recommender, because it is not guaranteed that important traits of the seed item are preserved. In our running example, the translated representation may seem like it belongs to a pair of pants, but it might not lead to good recommendations, as it is not constrained to retain either concrete or abstract properties of the original shirt like style, age group, or price.

To address this issue, the *reconstructor* is introduced, which encourages the *category translator* to be *cycle consistent* and to retain the traits of the seed item. Specifically, this network receives a translated representation v_i^c and the original category of the seed item. It then aims to reconstruct the original representation of the seed item v_i by returning a vector v_i' such that $v_i \approx v_i'$.

5.2 Model training

The adversarial learning component incorporates two types of loss functions: for the *reconstructor* and for the *classifier*. To allow end-to-end learning of the entire architecture, the final loss is given by a weighted sum of these two losses and the loss function of the supervised learning component. The weights for the losses constitute a convex combination and are determined by hyperparameters.

Cycle consistency loss. The loss of the *reconstructor*, in our approach is the Euclidean distance $\|v_i - v_i'\|^2$. Since both terms in the loss, v_i and v_i', are outputs of learned networks, a degenerated, yet optimal loss can be achieved. For instance, if the *item encoder* and the *category translator* return the same fixed value, regardless of the input, then the loss would be 0. To overcome this issue, we recall that the purpose of this loss is to motivate the *category translator* to not dismiss important traits in the seed item, represented by v_i. Therefore, v_i serves as a label for this loss. Consequently, we stop the gradients backpropagate from v_i due to this loss. We point out that this procedure allows to further improve the *item encoder*, due to the Jacobian obtained from v_i'.

Classifier loss. As mentioned above, given an item representation and a category, the *classifier* outputs the probability p that the represented item belongs to this category. Conventionally, its loss is cross-entropy and since it is supplied with a single true category, the loss is contracted to $-\log p$. We should bear in mind that only a well trained classifier can challenge the *category translator* and thereby allow it to generate suitable representations. This poses several difficulties and rules out the feasibility of a naive implementation of the architecture. We discuss these challenges and how we addressed them as follows. To optimally train the *classifier* we supply it with two types of training instances.

The first type consists of genuine outputs of the *item encoder*. Namely, we invoke the *classifier* twice, with the embeddings of the seed item v_s and the positive item v_p[6], where the label categories are the ones from the catalog. However, we note that a standard implementation would also affect the *item encoder*. Specifically, it can ultimately lead to a degenerated model, as the category is an input to the *item encoder*. That way, the *item encoder* is motivated to cooperate with the *category translator* so as to excel at this loss at the expense of the true objective of the model, which is supplying recommendations. For example, the *item encoder* may output the category embedding, while ignoring the rest of its input features. We therefore implemented the model in a way that we prevent the gradients of the *classifier* of flowing through the parameters of the *item encoder*. Our experiments revealed that this led to significantly improved performance.

As for the second type of input for the *classifier*, we recall that the *category translator* aims to create an embedding v_s^c, which makes the *classifier* classify v_s^c as if it belongs to category c. In order to avert a situation where the *category translator* finds edge cases (adversarial examples) that only deceive the *classifier*, but do not look like real vectors of items in category c, it is important that the *classifier* is trained on the output of the *category translator* v_s^c, with label category c. If done this way, v_s^c is used to train both the *category translator* and the *classifier*, but each of them has a different objective. The *category translator* aims to fool the *classifier*, while the latter needs to challenge the *category translator*. To address this issue, we use *adversarial training* with a gradient reversal layer. This means that during backpropagation the *category translator* obtains the original gradients, while the *classifier* is directed by the additive inverse of the gradients.

[6]For running time considerations the negative item's embedding is not fed to the *classifier*, although it could be trivially added.

We note that this model is scalable since its training time is linear in the size of the labeled set and item-category pairs; also, the amount of parameters grows linearly with the number of categories.

5.3 Specific CycleGAN Adaptations

Here, we lay out the commonalities and differences of CycleGAN and our approach in more detail. Specifically, this exposition will clarify how we transferred ideas from CycleGAN to the complementary item recommendation problem in an innovative way.

CycleGAN is a variant of Generative Adversarial Networks (GAN) [9], which includes two generators and two discriminators that are trained simultaneously. Let X be the source domain and Y the target domain for a given translation problem (e.g., summer to winter). Generator G learns to transform images from X to Y. Generator F learns to transform images from Y to X. Discriminator D_X learns to differentiate between genuine images of X and generated images $F(Y)$. That is, its objective is to return a high probability value for $x \in X$ and a low probability value for $\mathsf{F}(y)$, for $y \in Y$. Similarly, discriminator D_Y operates in domain Y. For simplicity, we collectively refer to D_X and D_Y as D, which represents a discriminator that distinguishes between genuine and generated images. Training of these networks is done using adversarial training. It is generally desired that $\mathsf{G}(x)$ does not dismiss important characteristics of the original image. This would guarantee that $\mathsf{G}(x)$ is a translation of x, rather than just an image that seems to belong to Y, but has no affinity to x. To this end, the cycle consistency loss is introduced. Namely, this loss asks to minimize the difference between x and $\mathsf{F}(\mathsf{G}(x))$.

We first explain that in a sense, the *item encoder* serves as the real distribution, the *category translator* as G, the *classifier* as D, and the *reconstructor* as F. Given an input representation v and a category c, the *classifier* works as follows. If v is a was generated directly by the *item encoder*, then it should confirm that the item belongs to the declared category c; otherwise, v was further translated by the *category translator*, and the *classifier* should reject an assumed affinity between the item and the category. Therefore, similar to GANs, the *classifier* (D) aims to output a probability close to one for inputs drawn from the *item encoder* (real distribution) and probability of zero for inputs from the *category translator* (G). Like in CycleGAN, our approach applies the cycle consistency loss using the *reconstructor*, which translates the item representation back to its original category and therefore serves as F.

However, there are some notable differences between the two models. First, the instances drawn from the real distribution in CycleGAN are genuine images from the source domain. Therefore, the cycle consistency loss is well defined, as the original image x serves as the label for $\mathsf{F}(\mathsf{G}(x))$. In contrast, our approach works on the feature space. That is, the "real distribution" is actually the output of the *item encoder*, which is a *learned* network. Consequently, the outputs of the *image encoder* also serve as labels, which might hamper its training. As mentioned before, we overcome this problem by stopping the gradients that arise from the seed item's vector.

Another difference stems from the complexity of the *category translator*. While in CycleGAN there are only two domains, in our problem the number of domains (categories) can reach to hundreds. Therefore we resort to training a unified translator for all categories.

Given the unified translator it might seem like the *reconstructor* is redundant, as also the *category translator* can translate between any pair of categories. However, in our experiments we noticed a crucial advantage in separating these two networks. We conjecture that it is due to their different objectives. The *category translator* aims to find good recommendations, while the *reconstructor* wishes to recover the original latent traits of the item. Therefore, the *category translator* may generate vectors that are similar to those of popular or high quality items, since they usually make good recommendations, while the *reconstructor* is not bound to this need. Therefore, the *category translator*, which generates the recommendations, should preferably be distinct from the *reconstructor*.

6 EXPERIMENTAL EVALUATION

We conducted an in-depth experimental evaluation of our approach. All our code is shared online for reproducibility [7].

6.1 Experiment Design

Datasets & Preprocessing. We rely on real-world data from *Amazon* [13], as used in previous related works [11, 24]. From these datasets, we first extracted the items' side-information, which include image, price, and category. Furthermore, each item i in the dataset can be associated with a list of recommended items $\{r_i\}$ that are *frequently bought together*. By removing items from $\{r_i\}$ that belong to the same category as i, we create a labeled set of complementary item pairs $\{(s, r_i)\}$, which we use for model training and evaluation in the experiments. We note that these pairs may be asymmetric, e.g., a laptop may be accompanied by a recommendation for a charger, but not vice versa.

We considered three subsets of the *Amazon* datasets: "Clothing, Shoes and Jewelry" (dubbed as Clothing), "Home and Kitchen" (Home), and "Toys and Games" (Toys). These subsets were chosen because complementary items are of key importance in these domains. At the same time, they differ in key aspects like their verticals, target users, attributes that affect the notion of complementary, and number of items and categories. The main statistics of these datasets are shown in Table 1.

Table 1: Dataset statistics

Statistic	Clothing	Toys	Home
#items	14,591	20,510	29,258
#item pairs	53,375	112,964	162,497
#categories	126	124	286
#category pairs	4,621	5,734	18,999
Avg. items per category	115.8	165.4	102.3
Max items per category	808	2,558	796
Min items per category	12	16	11

In terms of pre-processing, we excluded categories with less than five items to reduce noise. As a pre-trained image processing model, we used ResNet152 [12]. If an item had multiple images, we consider only the first one. We furthermore discretized the continuous price to twenty bins using equal-depth binning. We used a random sample of 80% of the data for training and the rest for validation and testing. To mimic the cold start problem, we ensured that the seed items in the validation and test sets do not appear in the training set.

Evaluation Procedure & Metrics. The output of our model is a ranked list of complementary items, given a seed item. We therefore apply standard list-based accuracy metrics, namely Hit Rate (*HR@k*) and NDCG (Normalized Discounted Cumulative Gain), see [33]. Moreover, since the labeled set is highly skewed to popular items, some models may focus on a narrow set of such popular items in their recommendations, thus leading to limited coverage and diversity. Therefore, we also report *catalog coverage* [15], which is defined as the fraction of the items of the catalog that appear in the top-k recommendation lists for the seed items in the test set. The value of k in this measure was 10.

Following the described problem setting, in the main evaluation protocol the desired category of the recommendations is given. To showcase the robustness of the methods, we also experiment in a setting where the recommendations can come from any category. We recall that our model requires a target category as an input. To create a recommendation list that considers all relevant categories, we go over all complementary categories of the seed item in a round-robin fashion and iteratively select the next recommendation.

Baselines. We compare our model against these baselines.

- **Popularity** is a simple yet effective baseline, which utilizes the labeled set to count the popularity of each item. For each item i it records the number of seed items for which i is labeled as their complementary item in the labeled set.
- **DCF** [8] is a recent neural model optimized for cold items. Since its code was not released, we implemented it based on the the original paper and make it publicly available.
- **DCF-Hard** is a variant of *DCF* that we propose here. It leverages category information to apply *hard negative mining*. Specifically, negative samples are not drawn from the entire catalog, but only from the items in the category of the labeled target item.
- **P-Companion** [11] is another recent neural model. Since its code is not publicly available, we implemented it and publish the code in our repository as well. To make it compatible with our problem setting we made two modifications. First, we control the target categories and second, we omit collaborative information to support the cold start problem.

We carefully tuned the hyperparameters both of our model and of the baselines for each of the datasets in a manual process. The final hyperparameters can be found in the code repository.

6.2 Results

6.2.1 Main Results. Table 2 reports the main results of our experiments, where the category is given to the recommenders. We name our proposed model **ALCIR**, which stands for ***A**dversarial **L**earning for **C**omplementary **I**tem **R**ecommendation*. To analyze the contribution of the individual elements of our architecture, we report results for (a) *ALCIR-Sup*, which only consists of the supervised learning component from Section 4 and (b) *ALCIR*, which corresponds to the *full* model as described in Section 5.

[7] https://fb.me/cgan_ccomplementary_item_recommendation

Table 2: Category-aware recommendation performance

Method	NDCG	HR@1	HR@5	HR@10	Cov. (%)
Clothing Dataset					
Popularity	0.238	0.051	0.153	0.234	6.99
DCF	0.211	0.011	0.059	0.111	7.24
DCF-Hard	0.226	0.019	0.083	0.147	40.49
P-Companion	0.238	0.030	0.102	0.169	84.59
ALCIR-Sup	0.298	0.074	0.203	0.302	**89.92**
ALCIR	**0.316**	**0.092**	**0.233**	**0.332**	88.5
Toys Dataset					
Popularity	0.231	0.038	0.126	0.200	5.13
DCF	0.193	0.009	0.042	0.081	5.96
DCF-Hard	0.208	0.017	0.064	0.111	36.72
P-Companion	0.235	0.028	0.104	0.172	**93.78**
ALCIR-Sup	0.297	0.073	0.205	0.303	90.46
ALCIR	**0.308**	**0.078**	**0.224**	**0.330**	82.62
Home Dataset					
Popularity	0.251	0.048	0.160	0.255	8.2
DCF	0.211	0.011	0.051	0.102	8.32
DCF-Hard	0.221	0.014	0.046	0.077	12.82
P-Companion	0.245	0.032	0.111	0.187	**94.45**
ALCIR-Sup	0.296	0.068	0.197	0.293	93.44
ALCIR	**0.304**	**0.077**	**0.210**	**0.312**	94.38

The results in Table 2 show that *ALCIR* consistently outperforms all baselines in terms of the accuracy measures on all three datasets, usually with a large margin[8]. We additionally observe that already the supervised component of our model (*ALCIR-Supervised*) performs better than the baselines. The adversarial component is then successful in even further increasing these already strong results.

Considering the ranking of the other models, we notice that the popularity-based approach represents a baseline that can be difficult to beat. The *DCF* model and even the improved *DCF-Hard* model never reach the accuracy levels of the popularity-based method. *P-Companion* works better, but still does not reach the Hit Rate values of popular-item recommendations. Only in terms of the NDCG, *P-Companion* reaches a similar performance level. The strong performance of the popularity-based method is not surprising, though. An inspection of the datasets revealed that the ten most popular items cover between a fifth and a quarter of the labeled complementary items. Moreover, also the evaluation of a "Co-Purchase" method in [11] showed that *P-Companion* did not outperform such a popularity-based approach in a user-centric study[9].

Our proposed model, in contrast, outperforms the popularity-based model consistently. We also observe that *ALCIR* leads to high *coverage* values, ranging from 82% to 94%. Only the *P-Companion* method reaches an even slightly higher catalog coverage. The popularity-based method by design only recommends very popular items, leading to the lowest coverage. Interestingly, also the *DCF* method has very low catalog coverage. The improved DCF version (*DCF-Hard*) helps to address the coverage problem to a good extent.

[8]We report results at additional cut-off thresholds in the code repository.
[9]Popularity-based recommendations were not examined in [8].

Table 3 finally shows the performance results when we consider all relevant categories for complementary item recommendations as described above. Naturally, across all models, the performance results for this experiment are lower compared to a case where the category is known. Nevertheless, we observe that the proposed model ALCIR is superior also in this problem setting.

Table 3: Recommendation performance w/o target category

Method	NDCG	HR@1	HR@5	HR@10	Cov. (%)
Clothing Dataset					
Popularity	0.099	0.001	0.005	0.007	0.06
DCF	0.111	0.000	0.001	0.002	0.09
DCF-Hard	0.118	0.001	0.003	0.007	1.82
P-Companion	0.111	0.000	0.002	0.003	0.62
ALCIR-Sup	0.150	0.009	0.036	0.060	59.36
ALCIR	**0.170**	**0.012**	**0.060**	**0.098**	**60.55**
Toys Dataset					
Popularity	0.102	0.001	0.003	0.005	0.04
DCF	0.124	0.000	0.001	0.002	0.06
DCF-Hard	0.148	0.003	0.010	0.016	5.67
P-Companion	0.123	0.001	0.001	0.002	5.12
ALCIR-Sup	0.172	**0.010**	0.044	0.021	**64.56**
ALCIR	**0.184**	0.009	**0.046**	**0.071**	56.03
Home Dataset					
Popularity	0.093	0.000	0.001	0.002	0.03
DCF	0.117	0.000	0.001	0.002	0.05
DCF-Hard	0.117	0.000	0.002	0.003	0.13
P-Companion	0.117	0.000	0.001	0.001	1.61
ALCIR-Sup	0.143	0.005	0.021	0.033	53.42
ALCIR	**0.150**	**0.008**	**0.030**	**0.049**	**55.53**

6.2.2 *Additional Analyses.*

Impact of Data Paucity. One main assumption when we introduced the unsupervised learning component to our model was that it will be particularly beneficial for rare categories. To validate this assumption, we conducted the following analysis to assess the relative contribution of the full *ALCIR* method when we vary the amount of labeled data for each pair of categories. To this end, we count the number of labeled instances per each pair of complementary categories and discretize these counts to ten equally-sized bins. We order the bins by the amount of labeled data for each pair of categories in ascending order. The first bins therefore represent pairs of complementary categories for which little supervision is available in the training set.

Figure 2 shows the performance of each model for the ten bins. We note that performance measures between different bins cannot be trivially compared, because each bin holds different target categories, with a different amount of items from which the recommender should select. Therefore, only the performance of the different models within the same bin and dataset can be compared, since they refer to the exact same test items. Considering this aspect, we can observe the same trend in all datasets, where the relative advantage of *ALCIR* is higher for the rare category pairs, i.e, in

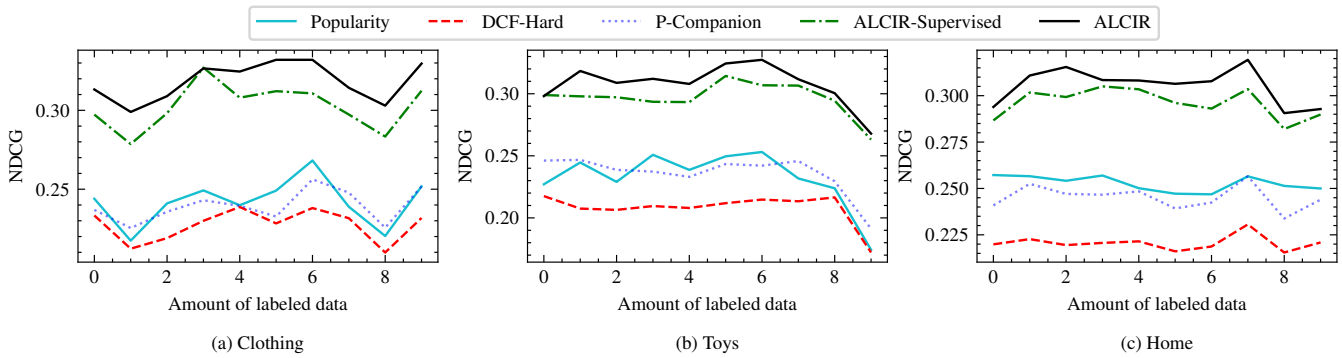

Figure 2: Performance with respect to the amount of labeled data

Table 4: Ablation study for category aware recommendation

Method	NDCG	HR@1	HR@5	HR@10
Clothing Dataset				
Classifier+Cycle	0.21	0.015	0.059	0.114
	(-33.5%)	(-83.9%)	(-74.7%)	(-65.7%)
ALCIR-Sup	0.298	0.074	0.203	0.302
	(-5.7%)	(-19.4%)	(-13.1%)	(-8.9%)
Triplet+Cycle	0.311	0.087	0.226	0.319
	(-1.8%)	(-5.6%)	(-3.1%)	(-3.8%)
Triplet+Classifier	0.305	0.082	0.213	0.306
	(-3.5%)	(-11.3%)	(-8.8%)	(-7.5%)
ALCIR	0.316	0.092	0.233	0.332
Toys Dataset				
Classifier+Cycle	0.207	0.016	0.07	0.124
	(-32.9%)	(-79.5%)	(-68.7%)	(-62.5%)
ALCIR-Sup	0.297	0.073	0.205	0.303
	(-3.7%)	(-6.4%)	(-8.7%)	(-8.2%)
Triplet+Cycle	0.294	0.072	0.205	0.303
	(-4.4%)	(-6.8%)	(-8.3%)	(-8.4%)
Triplet+Classifier	0.298	0.073	0.209	0.309
	(-3.3%)	(-6.6%)	(-6.6%)	(-6.5%)
ALCIR	0.308	0.078	0.224	0.330
Home Dataset				
Classifier+Cycle	0.218	0.014	0.068	0.127
	(-28.1%)	(-81.2%)	(-67.5%)	(-59.3%)
ALCIR-Sup	0.296	0.068	0.197	0.293
	(-2.6%)	(-11.4%)	(-6.1%)	(-6.1%)
Triplet+Cycle	0.298	0.069	0.201	0.303
	(-2%)	(-9.9%)	(-4.1%)	(-2.9%)
Triplet+Classifier	0.297	0.069	0.199	0.297
	(-2.1%)	(-10.7%)	(-4.9%)	(-4.7%)
ALCIR	0.304	0.077	0.210	0.312

the left parts of the figures. In contrast, the advantage of the full model becomes smaller for the very popular category pairs, and it is particularly pronounced for the Toys and Home datasets. Overall, the analysis confirms our assumption regarding the usefulness of the full model in particular for rare categories.

Ablation Study. Besides consisting of a supervised and an unsupervised component, one central feature of our architecture is that it makes use of a number of specific loss functions (triplet loss, cycle consistency loss and classifier loss), all of which are aimed to increase the performance of the model. To validate that these architecture elements contribute to the overall model performance, we ran an ablation study to assess the performance when only some of the components are utilized. Table 4 shows the outcomes of this analysis in absolute numbers and relative to the performance of the complete model (shown in parentheses). The results provide evidence that indeed all components contribute to the performance of the model. Most noteworthy is that a model that does not utilize labeled data, but only the unsupervised losses of the classifier and the cycle consistency performs the worst. This, however, comes as no surprise, given the well-known importance of labeled data.

7 FUTURE WORK

In this work we have proposed a novel semi-supervised approach for the highly relevant problem of complementary item recommendation, and an in-depth empirical evaluation clearly demonstrates the benefits of the approach. Our insights point to a number of future research directions. Our work focused on complementary item recommendations for cold items, and we assume that existing data (e.g., about co-purchases) can be used for the warm items. In future work, we plan to extend our model to also support warm items, using the same framework. That way, the input features for the item encoder would include also collaborative data by applying e.g., the method suggested in [4]. In addition to such extensions and continuing related research in [39, 46], other promising approaches to further improve the effectiveness of the model could lie in the *personalization* of the complementary item recommendations and to consider aspects of item *quality*. Finally, our problem setting can be thought as a special case of domain adaptation, where we transfer representations from one category to another. An interesting future work would be to extend our method to other recommendation scenarios, like context-aware recommendations, where we generate item representations in different contexts.

REFERENCES

[1] Unaiza Ahsan, Xiquan Cui, Rebecca West, Mingming Guo, Khalifeh Al Jadda, et al. 2020. Complementary Recommendations Using Deep Multi-modal Embeddings For Online Retail. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 1774–1779.

[2] Marina Angelovska, Sina Sheikholeslami, Bas Dunn, and Amir H Payberah. 2021. Siamese Neural Networks for Detecting Complementary Products. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*. 65–70.

[3] Tzoof Avny Brosh, Amit Livne, Oren Sar Shalom, Bracha Shapira, and Mark Last. 2022. BRUCE: Bundle Recommendation Using Contextualized item Embeddings. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 237–245.

[4] Koby Bibas, Oren Sar Shalom, and Dietmar Jannach. 2022. Collaborative Image Understanding. In *The 31st ACM International Conference on Information and Knowledge Management*.

[5] Yuri M Brovman, Marie Jacob, Natraj Srinivasan, Stephen Neola, Daniel Galron, Ryan Snyder, and Paul Wang. 2016. Optimizing similar item recommendations in a semi-structured marketplace to maximize conversion. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 199–202.

[6] Rami Cohen, Oren Sar Shalom, Dietmar Jannach, and Amihood Amir. 2021. A black-box attack model for visually-aware recommender systems. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 94–102.

[7] Lavinia De Divitiis, Federico Becattini, Claudio Baecchi, and Alberto Del Bimbo. 2023. Disentangling Features for Fashion Recommendation. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 19 (2023).

[8] Daniel A Galron, Yuri M Brovman, Jin Chung, Michal Wieja, and Paul Wang. 2018. Deep item-based collaborative filtering for sparse implicit feedback. *arXiv preprint arXiv:1812.10546* (2018).

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.

[10] Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1709–1719.

[11] Junheng Hao, Tong Zhao, Jin Li, Xin Luna Dong, Christos Faloutsos, Yizhou Sun, and Wei Wang. 2020. P-Companion: A Principled Framework for Diversified Complementary Product Recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2517–2524.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[13] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*. 507–517.

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

[15] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. *Transactions on Information Systems* 22, 1 (2004), 5–53.

[16] Cong Phuoc Huynh, Arridhana Ciptadi, Ambrish Tyagi, and Amit Agrawal. 2018. CRAFT: Complementary Recommendation by Adversarial Feature Transform. In *ECCV Workshops (3)*. 54–66.

[17] Dietmar Jannach and Michael Jugovac. 2019. Measuring the Business Value of Recommender Systems. *ACM Transactions on Management Information Systems* 10, 4 (2019).

[18] Dietmar Jannach, Oren Sar Shalom, and Joseph A Konstan. 2019. Towards More Impactful Recommender Systems Research.. In *ImpactRS@ RecSys*.

[19] Adit Krishnan, Hari Cheruvu, Cheng Tao, and Hari Sundaram. 2019. A modular adversarial approach to social recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1753–1762.

[20] Adit Krishnan, Ashish Sharma, Aravind Sankar, and Hari Sundaram. 2018. An adversarial approach to improve long-tail performance in neural collaborative filtering. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1491–1494.

[21] Sudhir Kumar and Mithun Das Gupta. 2019. c+ GAN: Complementary Fashion Item Recommendation. *arXiv preprint arXiv:1906.05596* (2019).

[22] Dokyun Lee and Kartik Hosanagar. 2019. How Do Recommender Systems Affect Sales Diversity? A Cross-Category Investigation via Randomized Field Experiment. *Information Systems Research* 30, 1 (2019), 239–259.

[23] Zhi Li, Bo Wu, Qi Liu, Likang Wu, Hongke Zhao, and Tao Mei. 2021. Learning the Compositional Visual Coherence for Complementary Recommendations. In *the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

[24] Yiding Liu, Yulong Gu, Zhuoye Ding, Junchao Gao, Ziyi Guo, Yongjun Bao, and Weipeng Yan. 2020. Decoupled Graph Convolution Network for Inferring Substitutable and Complementary Items. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*. 2621–2628.

[25] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of Session-based Recommendation Algorithms. *User-Modeling and User-Adapted Interaction* 28, 4–5 (2018), 331–390.

[26] Mansi Ranjit Mane, Stephen Guo, and Kannan Achan. 2019. Complementary-similarity learning using quadruplet network. *arXiv preprint arXiv:1908.09928* (2019).

[27] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring Networks of Substitutable and Complementary Products. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. 785–794.

[28] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. 43–52.

[29] Yehezkel S. Resheff, Yanai Elazar, Moni Shahar, and Oren Sar Shalom. 2019. Privacy and Fairness in Recommender Systems via Adversarial Training of User Representations. In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods - ICPRAM*. 476–482.

[30] Oren Sar Shalom, Shlomo Berkovsky, Royi Ronen, Elad Ziklik, and Amir Amihood. 2015. Data quality matters in recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 257–260.

[31] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 815–823.

[32] Oren Sar Shalom, Haggai Roitman, and Pigi Kouki. 2021. Natural language processing for recommender systems. In *Recommender Systems Handbook*. Springer.

[33] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. In *Recommender Systems Handbook*. Springer, 257–297.

[34] Xuemeng Song, Fuli Feng, Xianjing Han, Xin Yang, Wei Liu, and Liqiang Nie. 2018. Neural Compatibility Modeling with Attentive Knowledge Distillation. *CoRR* abs/1805.00313 (2018). http://arxiv.org/abs/1805.00313

[35] Christoph Trattner and Dietmar Jannach. 2019. Learning to Recommend Similar Items from Human Judgements. *User Modeling and User-Adapted Interaction* 30 (2019), 1–49.

[36] Ilya Trofimov. 2018. Inferring Complementary Products from Baskets and Browsing Sessions. *CoRR* abs/1809.09621 (2018). http://arxiv.org/abs/1809.09621

[37] Jian Wang, Badrul Sarwar, and Neel Sundaresan. 2011. Utilizing Related Products for Post-Purchase Recommendation in e-Commerce. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. 329–332.

[38] Rui Wang, Ning Yang, and S Yu Philip. 2022. Learning aspect-level complementarity for intent-aware complementary recommendation. *Knowledge-Based Systems* (2022), 109936.

[39] Tian Wang, Yuri M Brovman, and Sriganesh Madhvanath. 2021. Personalized embedding-based e-commerce recommendations at ebay. *arXiv preprint arXiv:2102.06156* (2021).

[40] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. 2018. A Path-Constrained Framework for Discriminating Substitutable and Complementary Products in E-Commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. 619–627.

[41] Jui-Chieh Wu, José Antonio Sánchez Rodríguez, and Humberto Jesús Corona Pampín. 2019. Session-based complementary fashion recommendations. *arXiv preprint arXiv:1908.08327* (2019).

[42] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Product knowledge graph embedding for e-commerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 672–680.

[43] An Yan, Chaosheng Dong, Yan Gao, Jinmiao Fu, Tong Zhao, Yi Sun, and Julian McAuley. 2022. Personalized complementary product recommendation. In *Companion Proceedings of the Web Conference 2022*. 146–151.

[44] Yuan Yao and F. Maxwell Harper. 2018. Judging Similarity: A User-Centric Study of Related Item Recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. 288–296.

[45] Wei Zhang, Zeyuan Chen, Hongyuan Zha, and Jianyong Wang. 2021. Learning from substitutable and complementary relations for graph-based sequential product recommendation. *ACM Transactions on Information Systems(TOIS)* (2021).

[46] Yin Zhang, Haokai Lu, Wei Niu, and James Caverlee. 2018. Quality-aware neural complementary item recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 77–85.

[47] Kui Zhao, Xia Hu, Jiajun Bu, and Can Wang. 2017. Deep style match for complementary recommendation. *arXiv preprint arXiv:1708.07938* (2017).

[48] Tong Zhao, Julian McAuley, Mengya Li, and Irwin King. 2017. Improving recommendation accuracy using networks of substitutable and complementary products. In *2017 International Joint Conference on Neural Networks (IJCNN)*.

[49] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International on Computer Vision*. 2223–2232.

[50] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. 2014. Bundle recommendation in ecommerce. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. 657–666.