Diversely-Supervised Visual Product Search

WILLIAM THONG and CEES G. M. SNOEK, University of Amsterdam, the Netherlands

This article strives for a diversely supervised visual product search, where queries specify a diverse set of labels to search for. Where previous works have focused on representing attribute, instance, or category labels individually, we consider them together to create a diverse set of labels for visually describing products. We learn an embedding from the supervisory signal provided by every label to encode their interrelationships. Once trained, every label has a corresponding visual representation in the embedding space, which is an aggregation of selected items from the training set. At search time, composite query representations retrieve images that match a specific set of diverse labels. We form composite query representations by averaging over the aggregated representations of each diverse label in the specific set. For evaluation, we extend existing product datasets of cars and clothes with a diverse set of labels. Experiments show the benefits of our embedding for diversely supervised visual product search in seen and unseen product combinations and for discovering product design styles.

CCS Concepts: • **Computing methodologies → Visual content-based indexing and retrieval;**

Additional Key Words and Phrases: Product retrieval, diverse supervision, diverse representation

ACM Reference format:
William Thong and Cees G. M. Snoek. 2022. Diversely-Supervised Visual Product Search. *ACM Trans. Multimedia Comput. Commun. Appl.* 18, 1, Article 13 (January 2022), 22 pages.
https://doi.org/10.1145/3461646

1 INTRODUCTION

The objective of this article is to retrieve specific images of products, such as cars or clothes. Searching for product images has a long tradition in computer vision and multimedia, covering query-by-instance [5, 23, 28, 40, 60], query-by-category [6, 9, 12, 16], query-by-attribute value [32, 47, 64, 74, 81], or query-by-description [27, 37, 68]. A more targeted search strategy has been proposed recently, in which a query-by-sentence aims to modify attribute values [1, 19, 66, 80] or to generate product instances [2, 82]. While these previous works consider the similarity of instance, category, and attribute labels individually, we aim to integrate them altogether to enable a more expressive product search.

We are inspired by recent works on diverse supervision [53, 73], which define auxiliary labels in separate branches to benefit a primary task. Ruder et al. [53] show the benefits of part-of-speech tagging as auxiliary labels for several natural language processing problems. Ye et al. [73] leverage image-level, box-level, and pixel-level annotations jointly for instance segmentation. Encouraged by these seminal works, we introduce diverse supervision to visual product search. We define the search for a given diverse set of labels as our primary task. To achieve this, we learn visual

Authors' address: W. Thong and C. G. M. Snoek, University of Amsterdam, Science Park 904, Amsterdam, the Netherlands; emails: {w.e.thong, cgmsnoek}@uva.nl.

1551-6857/2022/01-ART13 $15.00
https://doi.org/10.1145/3461646

representations for attribute, instance, and category labels altogether in an integrated embedding space for a product retrieval task.

Our main contribution is the introduction of diversely supervised visual product search, where the objective is to search for product images that match to a specific set of diverse labels. For example, we may want to retrieve images of "a *shirt* with *long* sleeves and a *stripe* print," which composes a set of three different labels. For this purpose, we derive an embedding space where interrelations among labels result in interrelated representations. Training relies on a diverse supervision of attribute, instance, and category labels to describe images through a diverse representation. For every label, we compute a representation by aggregating the corresponding items in the training set. We propose an evaluation based on composite queries for diversely supervised product search. We represent composite queries by averaging the aggregated visual representations of each diverse label of the specific set. As such, we create two diversely labeled datasets, which build upon existing clothes [40] and cars datasets [33, 72]. Evaluation on these two datasets shows the benefits of our embedding for diversely supervised product search in seen and unseen settings and for discovering the typicality effect of product styles. All source code and setups are released to foster further research in diversely supervised visual product search.[1]

2 RELATED WORK

2.1 Visual Product Search

Visual product search has attracted a lot of interest from social media platforms [25, 76, 78] or online e-retailers [71, 79], as they need to recommend products to users. In multimedia and computer vision applications, this interest in visual product search has been translated into different retrieval problems. Each problem comes with its own challenge and offers new ways to search for products.

One line of work follows the traditional instance retrieval problem where an example image is used as a query. The objective is to retrieve images of the same product in a gallery set within the same domain [40, 60] or across domains [5, 23, 28, 38, 40]. Product categories can also be related to each other to retrieve complementary products for recommendation by capturing a global description of style [22, 29, 45, 65].

Another line of work covers image captioning where a description is matched to an image [27, 37, 68]. The idea is to learn a multimodal embedding where text and image representations are aligned together [27]. Grounding words in the image is particularly important to capture the interactions between both modalities [37, 68]. In this article, the search task is complementary to text-image retrieval, as we consider an unordered set with a varying number of labels instead of a fixed description sentence.

Finally, another line of work explores relevance feedback to integrate input from the user. This can consist of a comparison of product pairs to assess the relative strength of attributes [32, 47, 74], to verify that they exhibit the same attribute value [64, 81], or indicate a location of attribute interest [24]. Alternatively, the user can manipulate one attribute value to retrieve [1, 19, 66, 80] or to generate [2, 82] the targeted product. In this article, we introduce a complementary problem: we search for products that match to a specific, yet diverse, set of labels.

2.2 Diverse Labels

Searching for a diverse set of labels has mainly focused on describing images with multiple binary attributes. Multi-attribute queries are used to search for images of faces [34, 56, 58] by describing

[1] https://github.com/twuilliam/diverse-search.

the absence or presence of facial traits. The conjunction of positive binary attribute values has also proven to be useful in animal categorization, in a retrieval setting [49] or a zero-shot classification setting [3, 14, 35]. While attributes are important to describe objects, they are not specific enough for producing a product search [15]. Different from these works, we aim to learn (a) image similarities through a diverse set of labels that go beyond attributes by including category and instance labels; and (b) an embedding space that encodes every label with real-valued vector representations rather than binary representations.

Structured queries have also been proposed to capture a diverse set of relations for complex scene retrieval. Sentence queries go beyond simple keywords to capture relations among objects [17, 55, 66, 67]. Graph queries structure explicitly these relations [8, 26, 36]. Paragraph queries enable the retrieval of an image sequence to illustrate a story [30, 50]. In this work, we rely on a diverse set of label vocabularies to structure product retrieval. We form composite query representations by averaging over the visual representations of the desired labels to search for.

2.3 Diverse Representations

Encoding multiple labels into an embedding space is usually done through two different approaches. One approach is to learn a global representation of images [40, 72] to classify categories and attribute values. An alternative approach is to learn a subspace for each attribute to create distinct and disentangled similarities [64]. Variants of this approach enhance the backbone network to modulate channels either with a learned real-valued vector to promote constructive interference [81] or by a fixed binary mask to model task relationships in a non-parametric manner [62]. Yet, these approaches are restricted to comparing attribute [64, 64] or instance [40, 72] labels. In this article, we propose to encode attribute, instance, and category labels in an integrated manner by explicitly establishing their interrelationships.

3 METHOD

3.1 Problem Statement

During the training, we are given a training set of product images $\mathcal{X}_{train}$. Each image $\mathbf{x}$ in the training set comes along with a diverse set of labels. In particular, we are interested in the category label $y \in C$, the label v of attribute $k \in \mathcal{A}_k$, and the instance label $i \in \mathcal{I}_{train}$. C is the category vocabulary of C product categories. As products can express multiple attributes, we consider K different attribute vocabularies $\mathcal{A}_k$ with A_k attribute values each. Hence, images also have multiple attribute labels, forming multiple tuples (k, v) with $k = 1, \ldots, K$ and $v = 1, \ldots, A_k$. $\mathcal{I}_{train}$ is the set of instances in the training set. Instances are an integral part of visual products. Images of the same instance usually differ by a different viewpoint or background. Hence, the instance label enforces images of the same product to be close to each other. Overall, we leverage all $\{C, \mathcal{A}_1, \ldots, \mathcal{A}_K, \mathcal{I}_{train}\}$ labels to provide a diverse supervisory signal to the model during training.

During the evaluation, we are given a gallery set of images $\mathcal{X}_{gal}$, which originates from a separate set of products. Formally, $\mathcal{I}_{train} \cap \mathcal{I}_{gal} = \emptyset$. The gallery set $\mathcal{X}_{gal}$ shares the same category vocabulary C and K attribute vocabularies $\mathcal{A}_k$ with the training set $\mathcal{X}_{train}$. As such, these vocabularies serve to build a set of labels for describing composite queries used for retrieving product images. An example of such a search is to retrieve clothes images that match "a *shirt* with *long* sleeves and a *stripe* print," where the set of labels comprises one product category and values for two different attributes. Separating the instances in the gallery set $\mathcal{X}_{gal}$ from the training set $\mathcal{X}_{train}$ allows to evaluate the generalization ability of the model on new products that express both seen and unseen combinations of categorical and attribute values.

3.2 Diversely Supervised Embedding

We propose to learn a diversely supervised embedding space where Euclidean distances capture label similarities. The embedding space is motivated by the definition of attribute, instance, and category for describing products: (a) products are instances of particular categories and (b) attributes characterize visual properties of products. For example, a "3-Series sedan" is an instance of the "BMW" car category with "4 doors" and "5 seats" attributes. In this context, attribute and instance labels are highly interrelated to each other because attributes qualify instances. Our technical contribution lies in how to explicitly encode these label definitions in the diversely supervised embedding space.

To learn a representation for each label, we rely on a cross-entropy loss with softmax embedding [39, 46, 59]. While originally proposed for either instance retrieval [39, 46] or few-shot learning [59], we develop a variant for learning a representation from a diverse set of labels. Different from the commonly used contrastive [11, 18] or triplet [57, 69] losses, the proposed loss does not require any intricate sampling, which makes the training with diverse supervision much simpler. We derive below how to learn representations for each label type in the embedding space.

Attribute representations. We encode attribute labels in subspaces, one per attribute. A dataset with K attributes results in an embedding with K subspaces. Let $\mathbf{h} = f_\theta(\mathbf{x})$ be the features $\mathbf{h}$ of an image $\mathbf{x}$ from a convolutional network f with trainable parameters θ. The idea is to learn a linear projection of the features $\mathbf{h}$ in multiple separate subspaces to encode the representation for each attribute $k = 1, \ldots, K$ in a representation $\mathbf{z}_{A_k} \in \mathbb{R}^d$:

$$\mathbf{z}_{A_k} = \mathbf{W}_k \mathbf{h} + \mathbf{b}_k, \tag{1}$$

where $\mathbf{W}_k$ and $\mathbf{b}_k$ are the weights and biases, respectively. We learn the attribute representation based on the cross-entropy loss with softmax embedding:

$$\mathcal{L}_{A_k} = -\log \frac{\exp(-\|\mathbf{z}_{A_k} - \mathbf{a}_{k,v}\|)}{\sum_{z \in \mathbb{Z}_{A_k}} \exp(-\|\mathbf{z}_{A_k} - \mathbf{a}_{k,z}\|)}, \tag{2}$$

where $\|\cdot\|$ is the Euclidean distance, $\mathbb{Z}_{A_k}$ denotes the set of all the latent prototypes $\mathbf{a}_{k,v} \in \mathbb{R}^d$ of attribute k. The softmax embedding function provides a probability of the attribute representation $\mathbf{z}_{A_k}$ to be recognized as the value v of attribute k. At each step, the model pulls $\mathbf{z}_{A_k}$ to its corresponding latent prototype $\mathbf{a}_{k,v}$, and pushes it away from the prototypes of other values $\mathbf{a}_{k,z}$.

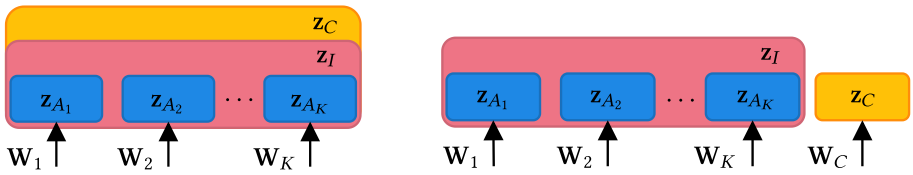
Instance representations. We establish an interrelation between instance and attribute representations. As attribute labels qualify product instances, we encode this property in the embedding space. The instance representation $\mathbf{z}_I \in \mathbb{R}^D$ with $D = K \cdot d$ corresponds to the concatenation of all attribute subspaces:

$$\mathbf{z}_I = \bigcup_{k=1}^{K} [\mathbf{z}_{A_k}], \tag{3}$$

where $\bigcup[\cdot]$ is the vector concatenation operator. Similarly, we learn the instance representation based on the cross-entropy loss with softmax embedding:

$$\mathcal{L}_I = -\log \frac{\exp(-\|\mathbf{z}_{I_k} - \mathbf{p}_i\|)}{\sum_{z \in \mathbb{Z}_I} \exp(-\|\mathbf{z}_{I_k} - \mathbf{p}_z\|)}, \tag{4}$$

where $\mathbb{Z}_I$ denotes the set of all the latent instance prototypes $\mathbf{p}_i \in \mathbb{R}^D$.



(a) *Triple* grouping (b) *Dual* grouping

Fig. 1. Diversely supervised embeddings. We consider attributes (*blue*), category (*yellow*), and instance (*pink*) representations. Given the output features of a convolutional network, we learn multiple linear projections $\mathbf{W}$ to an embedding space. (a) The *triple* grouping makes the embedding axis-aligned on attributes for both instances and categories. Average representations of instances form category representations. (b) The *dual* grouping treats category representations in a separate subspace.

Category representations. We propose two different variants to encode the category labels, as illustrated in Figure 1: (a) the *triple* grouping ensures that representations from instances of the same category are close to each other (Figure 1(a)), while (b) the *dual* prefers to encode the category labels separately (Figure 1(b)). We also learn the category representation based on the cross-entropy loss with softmax embedding:

$$\mathcal{L}_C = -\log \frac{\exp(-\|\mathbf{z}_C - \mathbf{c}_y\|)}{\sum_{z \in \mathbb{Z}_C} \exp(-\|\mathbf{z}_C - \mathbf{c}_z\|)}, \tag{5}$$

where $\mathbb{Z}_C$ denotes the set of all latent category prototypes $\mathbf{c}_y$. In the *triple* grouping, category representations are a concatenation of attribute subspaces $\mathbf{z}_C \in \mathbb{R}^D$. Hence, $\mathbf{z}_C$ is also a concatenation of a series of $\mathbf{z}_{A_k}$, just like $\mathbf{z}_I$. Though, we impose a constraint on Z_C such that grouping instance representations form category representations. In other words, without the loss on $\mathbf{z}_C$ instance representations would be free to organize themselves in the embedding space. Formally, the category representation corresponds to:

$$\mathbf{c}_y = \frac{1}{|\mathbb{Y}|} \sum_{i \in \mathbb{Y}} \mathbf{p}_i, \tag{6}$$

where $\mathbb{Y}$ is the set of all latent instance prototypes of the category y. In the *dual* grouping, they are linearly projected to their own subspace $\mathbf{z}_C \in \mathbb{R}^d$.

The grouping motivation differs by the assumptions on how to relate instance, category, and attribute labels. We assume that attributes qualify instances, and categories emerge by grouping instances. This leads to the *triple* grouping, where all three types of labels are interrelated. The *dual* grouping relaxes the category assumption by only interrelating attributes and instances. The former incorporates the fact that categories and instances play opposite roles: Categories force the embedding to be agnostic to instances, while instances force the embedding to focus on fine-grained differences making categories harder to learn.

Training. The training objective of the diversely supervised embedding corresponds to a minimization of a weighted sum of representations for each type of labels:

$$\mathcal{L} = \lambda_I \mathcal{L}_I + \frac{\lambda_A}{K} \sum_k \mathcal{L}_{A_k} + \lambda_C \mathcal{L}_C + \lambda_R \|\mathbf{z}\|^2, \tag{7}$$

where λ_I, λ_A, and λ_C denote tradeoff hyperparameters to control the contribution of each type of label. Some images might not express all attributes K defined in the dataset, e.g.,, a *skirt* does not have a *sleeves length* attribute. In this case, the contribution of the missing attribute in Equation (7)

is ignored. We also apply an ℓ_2 regularization on the final representation $\mathbf{z}$, which encodes all label types. In the *triple* grouping, the final representation is $\mathbf{z} = \mathbf{z}_I \in \mathbb{R}^D$ while in *dual*, $\mathbf{z} = [\mathbf{z}_I; \mathbf{z}_C] \in \mathbb{R}^{(K+1)\cdot d}$.

Prototype updates. To design the probabilistic model, we take inspiration from the prototype literature [39, 46, 59], where the general idea is to apply a softmax over distances to prototypes. Different from prototypical networks [59], we consider prototypes as latent parameters, which are initialized randomly and updated throughout the training like any other neural network parameters. In other words, the backward pass also includes the partial derivative of the loss with respect to all latent prototypes. This differentiates us from prototypical networks [59]. Indeed, rather than defining prototypes as the average of support image representations, our prototypes are latent representations that are updated during training. Compared to a classification setting [59], no support images are present in retrieval, which is why we design prototypes as latent representations as usually done in instance retrieval [46, 77].

Implementation details. The backbone network relies on ResNet50 [21] pre-trained on ImageNet [54]. To produce the embedding space, the classification layer is removed and replaced by the multiple linear projections with a random weight initialization. Latent prototypes are also initialized with random weights. During training, the model minimizes the loss function described in Equation (7) using the Adam stochastic optimizer algorithm [31]. Images are cropped given their bounding box labels and resized to 224×224 and augmented with horizontal flipping. Hyper-parameters are the following: minibatch size of 128, learning rate of $1e{-}4$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay of $5e{-}5$, and subspaces are of size $d = 50$. We set the tradeoffs to $\lambda_I = \lambda_C = \lambda_A = 1$ and $\lambda_R = 1e{-}3$. Updates of the latent prototypes operate at a learning rate $10\times$ higher. The learning rate undergoes a cosine annealing decay without restart [41]. We set hyper-parameters according to the classification accuracy of attributes on the validation set. The implementation relies on the PyTorch framework [48].

3.3 Composite Queries Representations

During the evaluation, we query the gallery set $\mathcal{X}_{gal}$ with composite queries derived from the training set. We represent composite queries by a real-valued vector $\mathbf{q} \in \mathbb{R}^D$ of M diverse labels. In other words, given a composite query $\mathbf{q}$, the idea is to retrieve product images in the gallery set $\mathcal{X}_{gal}$ from their visual representations $\mathbf{z}$ that match a specific set of M labels. To form composite query representations, we average the representations from the training set of each $m \in M$ label individually and take the overall average. Formally, this corresponds to a per-label averaging:

$$\mathbf{q} = \frac{1}{M} \sum_{m=1}^{M} \frac{1}{|\mathcal{M}_m|} \sum_{n \in \mathcal{M}_m} \mathbf{z}^{(n)}, \tag{8}$$

where $\mathcal{M}_m$ is the set of training images that exhibits label m with $m = 1, \ldots, M$. The inner sum averages the representations $\mathbf{z}$ of all images $n \in \mathcal{M}_m$ for each label m. The outer sum calculates an average of averages to create a composite query representation $\mathbf{q}$ that includes all M labels. If normalization is done globally (i.e., moving $1/|\mathcal{M}_m|$ to the outer sum), it corresponds to a per-sample averaging.

4 EVALUATING DIVERSELY SUPERVISED VISUAL PRODUCT SEARCH

4.1 Diversely Labeled Datasets

We introduce two datasets for diversely supervised visual product search: *Diverse–Cars* and *Diverse–Clothes*. Both datasets include instance, category, and multiple attributes labels.

Instance: 3 Series sedan
Category: BMW
Attributes
·*Doors*: 4
·*Seats*: 5
·*Type*: sedan

(a) **Diverse–Cars**

Instance: 6096
Category: Jacket
Attributes
·*Fabric*: denim
·*Front*: buttons
·*Hemline*: n.a.
·*Neckline*: hood
·*Print*: distress
·*Shoulder*: n.a.
·*Sleeves*: long
·*Silhouette*: cropped

(b) **Diverse–Clothes**

Fig. 2. Diversely labeled examples from Diverse–Cars and Diverse–Clothes.

Diverse–Cars. We build upon Cars196 by Krause et al. [33] and CompCars by Yang et al. [72] to create Diverse–Cars. The original datasets intend to tackle fine-grained categorization and verification; we merge them for the task of diversely supervised product search. This creates a dataset that covers car models sold in both North American and South Pacific regions. We manually annotate Diverse–Cars to merge car model duplicates and to provide clean annotations for car makers and car attributes. Diverse–Cars defines 97 car makers and 3 car attributes. Every attribute is further defined with the specific values: 4 *number of doors*, 4 *number of seats*, and 12 *type* values. In total, Diverse–Cars contains 28,423 images from 386 car models for training and 22,450 images from 305 separate car models for evaluation.

Diverse–Clothes. We build upon In-Shop Clothes by Liu et al. [40] to create Diverse–Clothes. The original dataset provides a large number of clothing products along with multiple views and a rich description of several sentences, but the provided labels are known to contain scarce attribute values, duplicates, and incoherencies [75]. Hence, we manually re-annotate the dataset to provide clean annotations for clothes categories and clothes attributes. Diverse–Clothes defines 12 clothes categories and 8 clothes attributes. Every attribute comes with specific attribute values: 6 *fabric*, 7 *frontal feature*, 6 *hemline*, 13 *neckline*, 15 *print*, 4 *shoulder line*, 6 *sleeves length,* and 2 *silhouette* values. In total, Diverse–Clothes contains 25,862 images from 3,996 fashion products for training and 26,797 images from 3,982 separate fashion products for evaluation.

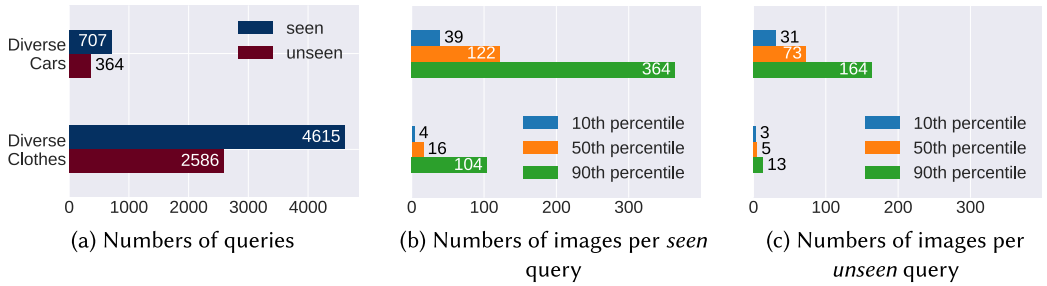
The annotation protocol is described in the Appendix. Data splits and diverse labels will be released. Figure 2 illustrates some diversely labeled examples for each dataset.

4.2 From Diversely Labeled Images to Composite Queries

We leverage composite queries to retrieve images in the gallery set $\mathcal{X}_{gal}$ that share the same set of labels. In this article, we define a composite query as a composite between a category label and one or multiple attribute labels, for a total of M types of label. An example of composite query for cars can be "a *BMW* with *2* doors, *5* seats and with a *coupé* type" ($M = 4$) while an example for clothes can be "a *shirt* with *long* sleeves and a *stripe* print" ($M = 3$). To avoid searching a needle in a haystack, we limit the number of attribute labels in composite queries to a maximum of three.

During the evaluation, we separate *seen* from *unseen* composite queries. If there is at least one image in the training set $\mathcal{X}_{train}$ that corresponds to the composite query, then the composite query is identified as *seen*. If the combination of category and attributes does not exist in the training set $\mathcal{X}_{train}$, then the composite query is *unseen*. Unseen composite queries are more realistic and more challenging than seen composite queries, because their combination has never been encountered by the model during training.

For each dataset, we generate composite queries by considering all possible category and attribute combinations and select the valid ones. A query is valid if there is at least one image in the gallery set $\mathcal{X}_{gal}$ with this specific combination. Figure 3 presents the distribution of seen and unseen composite queries for both datasets. Unseen composite queries constitute more than



(a) Numbers of queries (b) Numbers of images per *seen* query (c) Numbers of images per *unseen* query

Fig. 3. Composite queries distribution per dataset. Unseen queries represent one-third of all queries. Diverse–Clothes appears to be more challenging than Diverse–Cars given the very few images per query (in both *seen* and *unseen* scenarios).

one-third of the total queries, which illustrates the difficulty of both benchmarks. Diverse–Clothes provides a more challenging evaluation than Diverse–Cars. Indeed, there is a high number of queries and very few images per query considering the large gallery size. For example, given an unseen query, the median number of images per query is only 5, while the the gallery size is 26,797 images. Searching for new clothes products emerges as a more difficult task than searching for new car models, given the large diversity of fashion items.

4.3 Evaluation

Evaluation is performed on the gallery set $\mathcal{X}_{gal}$ that contains separate instances from the training set $\mathcal{X}_{train}$, as defined in Section 3.1. In other words, while a diverse set of labels defined by the composite query might have been *seen* or *unseen* during training, instances in the gallery have never been seen before. This is the protocol commonly used in zero-shot instance retrieval (e.g., References [40, 60]), where no overlap exists in terms of images nor instances between the training and the gallery sets. An ℓ_2 normalization is applied to the representations before measuring distances between the composite query $\mathbf{q}$ and the gallery $\mathcal{X}_{gal}$. A retrieved image is considered as a hit if it shares the set of labels with the composite query. We report the **mean average precision (mAP)** [44] across *seen*, *unseen* combinations for composite queries, and the *overall* to measure the performance.

5 RESULTS

5.1 Comparison with Alternatives

We adapt four existing methods, designed for a different purpose, in such a way that they become applicable to our setting. For fair comparisons, we apply the same procedure on these alternative models for both training and evaluation. We also use the same similarity loss based on the softmax embedding loss with prototypes. Below, we detail how each selected method is repurposed:

- **Global** maps an image $\mathbf{x}$ to a global representation $\mathbf{h}$, of the same dimension as our model with partial grouping. Inspired by Liu et al. [40], we add common softmax classification heads on top of the global embedding space to predict values for every K attributes and for categories. In other words, this corresponds to a multitask model with multiple heads. An additional similarity loss on the global embedding space models instance representations. In our setting, the final embedding used for evaluation is the global representation space.
- **Conditional** gives every label its own metric subspace $\mathbf{x}$, as originally introduced for attributes by Veit et al. [64]. Compared with our proposed method, conditional does not include any grouping mechanism. We add for every subspace label a loss that measures image

Table 1. Comparison with Alternatives

(a) **Diverse–Cars**

Method	seen	unseen	all
Global	33.20	19.45	28.53 ± 0.20
Conditional	33.83	19.83	29.00 ± 0.13
Modulation	32.80	17.57	27.63 ± 0.46
Routing	29.87	16.02	25.17 ± 0.14
This article	**37.61**	**21.03**	**31.98** ± 0.30

(b) **Diverse–Clothes**

Method	seen	unseen	all
Global	8.58	4.15	6.88 ± 0.08
Conditional	8.01	3.60	6.33 ± 0.13
Modulation	8.61	4.12	6.89 ± 0.08
Routing	6.61	2.56	5.06 ± 0.11
This article	**9.67**	**4.56**	**7.72** ± 0.13

We adapt four existing methods, designed for a different purpose, in such a way that they become applicable to our setting (details provided in Section 5.1). We report the average over three runs. Our embedding outperforms these alternatives in mAP (in %) on both Diverse–Cars and Diverse–Clothes datasets. Integrating attribute, instance, and category representations altogether in the embedding space with interrelated representations helps to model a diverse set of labels.

similarities. In our setting, the final embedding used for evaluation concatenates attribute, category, and instance subspaces.

- **Modulation** controls the amount of feature sharing for every type of labels, as originally proposed for attributes by Zhao et al. [81]. Similar to conditional, every type of label representation is also delimited to its subspace. Though, the main difference with conditional lies in the backbone network, which produces different features per label. Instead of having an explicit subspace per label during training, the idea is to encode the label information by transforming the activations of the backbone with a learned real-valued vector to weight every channel. This offers a compelling and efficient way to have label-specific feature representations without the need to train label-specific models. Following Zhao et al. [81], modulation occurs after the last two residual blocks (i.e., block3 and block4). In our setting, the final embedding concatenates the modulated attribute, category, and instance representations.
- **Routing** zeroes out channels given a type of labels, as originally proposed for many task learning by Strezoski et al. [62]. Routing is in the same spirit as modulation, and the difference lies in the usage of fixed binary masks to transform the activations of the backbone rather than learned real-valued vectors. Following Strezoski et al. [62], we generate binary masks by sampling a binomial distribution with a probability of success of 0.6. Similar to modulation, we apply the routing module after the last two residual blocks. In our setting, the final embedding concatenates the routed attribute, category, and instance representations.

Results on Diverse–Cars. Table 1(a) shows that our diversely supervised embedding outperforms alternative ways to combine attribute, category, and instance subspaces. Interestingly, channel-modulated methods based on a real-valued or binary masks achieve a lower retrieval score than the non-modulated conditional counterpart. As cars depict clear attribute values, their representation does not really benefit from creating a feature weighting. Indeed, there is no middle ground between three and four doors, while there might exist a debate to decide whether the sleeves length is *long* or *three-quarter*. When comparing with the conditional embedding, the diversely supervised embedding shows a large improvement. Integrating attribute, instance, and category representations altogether in the embedding space, rather than separating them all, helps to capture the diverse set of labels needed for diversely supervised search.

Note that for fair comparison, we implement alternatives with the same prototype loss as our method. For example, the conditional alternative of Veit et al. [64] has been initially proposed with a triplet loss. When training conditional with a triplet loss [57], the mAP drops by 9.74% on

Table 2. Per-sample *vs.* Per-label Averaging on Diverse–Cars

Averaging	seen	unseen	all
Per-sample	6.44	5.06	5.97
Per-label	**35.17**	**19.28**	**29.77**

Weighting per-sample biases in the query, which degrades the mAP (in %) score.

Table 3. Triple *vs.* Dual Grouping on Diverse–Cars

Grouping	seen	unseen	all
Triple	35.17	19.28	29.77
Dual	**38.10**	**20.71**	**32.19**

Separating the category representation leads to an mAP (in %) improvement.

Diverse-Cars. As triplets only capture one label at a time, results degrade in a multiple labels setting. Our proposed loss with latent prototypes allows us to capture all labels simultaneously, which results in an increased performance for the alternatives and our proposed model.

Results on Diverse–Clothes. Table 1(b) confirms the benefits of the diversely supervised embeddings on this more challenging dataset. When products exhibit more subjective attribute values, modulation has an edge over the non-modulated conditional counterpart. The routing module struggles the most, as zeroing out channels destroys information needed when measuring distances in the embedding space. When comparing the inference time, we notice the channel modulated methods have a linear complexity to the number of subspaces, as every label comes with a modulated representation. This is different from global, conditional, and ours that have a constant complexity, as they do not need to be channel-modulated. Our integration of attribute, instance, and category representations in the embedding space also captures these more subtle attribute changes without the need to modulate the backbone.

5.2 Ablations

Per-sample vs. *per-label averaging.* We study two alternatives to represent composite queries in the embedding space, as defined in Equation (8). Recall that we collect all visual representations corresponding to every label and average them either per-sample or per-label to form a representation for composite queries. Table 2 shows that a per-label averaging outperforms a per-sample averaging on Diverse–Cars. When averaging per-sample, all sample images are considered equally in the composite query. If a label is over-represented in the training set, then a per-sample averaging will result in a composite query biased towards this dominant label. When averaging per-label, all labels are instead considered equally. If a label is over-represented in the training set, then a per-label averaging will mitigate the imbalance effect, as an equal weight is put to each label representation to produce the composite query. For the remaining experiments, we then rely on a per-label averaging for composite queries to avoid a strong bias towards the dominant label.

Triple vs. *dual grouping.* In this experiment, we evaluate the difference between the triple and dual grouping in the embedding (Figure 1). The grouping motivation differs by the assumptions on how to relate instance, category, and attribute labels; and the practical application. In the triple grouping, attributes qualify instances, and grouped instances form categories. With all three types of labels interrelated in one single embedding space, this allows to explore the dataset to discover trends, as illustrated in Figure 9. The dual grouping relaxes the category assumption, as categories are now in a separate subspace. This avoids the duality where the embedding focuses on fine-grained instance differences while trying to group them for form categories at the same time. Table 3 shows that the dual variant outperforms the triple one on Diverse–Cars. A competing duality appears between instances and categories: Focusing on categories pushes the embedding to be agnostic to instances differences. The triple variant allows an interrelated exploration of products,

Table 4. Pre-training Comparison on Diverse–Cars and Diverse–Clothes

Pre-training	Cars	Clothes
Self-supervised	16.21	3.53
ImageNet	**32.19**	**7.74**

Pre-training on ImageNet improves by a factor two on the diverse search of all queries (mAP, in %) compared with a self-supervised pre-training. ImageNet acts as a regularizer.

Table 5. Swapping Backbones between Diverse–Cars and Diverse–Clothes

Swapping	Cars	Clothes
✓	0.86	0.33
	32.19	**7.74**

When swapping the backbones, the diverse search of all queries yields a very low performance (mAP, in %). Backbone features are specific to each dataset.

Table 6. Search Space Comparison on Diverse–Cars and Diverse–Clothes

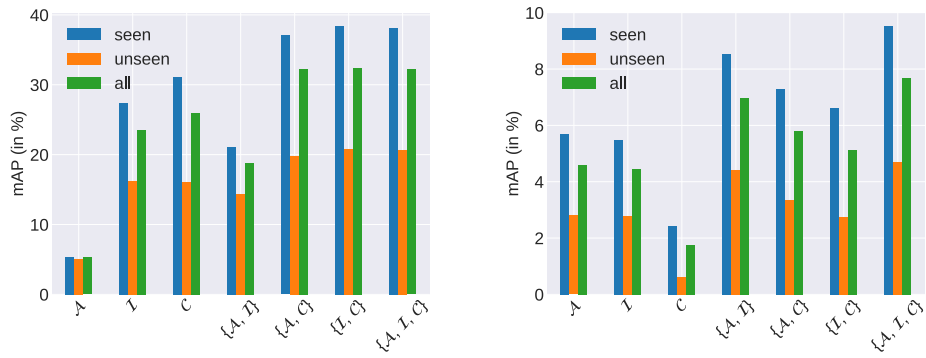
Search space	Model	Fine-tuning	Cars	Clothes
Features	Self-supervision		1.35	0.45
Features	Pre-trained		0.91	0.59
Features	Pre-trained	✓	22.02	3.64
Embedding	Pre-trained	✓	**32.19**	**7.74**

Fine-tuning on the respective datasets yields a significant mAP (in %) improvement over models trained in a supervised or self-supervised setting on ImageNet. Diversely supervised search benefits significantly when the search occurs in the embedding space, which captures all label types as opposed to the feature space.

as all diverse label representations are axis-aligned. Yet, putting the category representations in another subspace better helps the diversely supervised search. Additionally, we evaluate a variant where category labels are treated like any other attribute labels. In this variant, we obtain a 29.34% mAP. This reinforces the observation that category labels are then different from attributes and need to be treated accordingly. Depending on the application, it can be advantageous to separate instance and category representations. For the remaining experiments in this section, we use the dual grouping, as it yields the best scores for both seen and unseen queries.

Pre-training. We explore the effect of self-supervised pre-training on our model. We rely on MoCo v2 [10, 20] for the self-supervision training and use the same hyper-parameters as proposed originally. Once trained, we use these weights to initialize the ResNet50 backbone of our model. Table 4 compares a pre-training on ImageNet [54] with self-supervision on both Diverse-Cars and Diverse-Clothes. On both datasets, pre-training on ImageNet outperforms a pre-training with self-supervision. During training, we notably observe an overfitting effect with models initialized with self-supervision. Indeed, the training set of both datasets is several orders of magnitude smaller than ImageNet. Thus, a pre-training on ImageNet acts as a regularizer to help models generalize to diversely supervised search.

Swapping backbones. To understand the importance of backbone features in the generalization performance on diversely supervised search, we swap the backbone network trained on Diverse–Cars with the one trained on Diverse–Clothes, and vice versa. Concretely, $\mathbf{h}$ in Equation (1) for Diverse–Cars comes from the backbone f_θ of Diverse–Clothes, and vice versa. Table 5 shows the negative effect of swapping backbones. In either scenario, swapping the backbone drops the performance close to zero. This means that the backbone features, as well as the linear projections, are dataset-specific, as they cannot generalize across datasets.



(a) **Diverse–Cars** (b) **Diverse–Clothes**

Fig. 4. Influence of diverse labels. In both datasets, the instance supervision $\mathcal{I}$ is essential. For Diverse–Cars, the category supervision $\mathcal{C}$ matters the most, while for Diverse–Clothes the attribute supervision $\mathcal{A}$ is the most important. A combination of supervision results in an improvement for both seen and unseen composite queries.

Search space. We assess the importance of the embedding space for diversely supervised product search by comparing with a search in the feature space. Concretely, we compute the composite query representation in Equation (8) from $\mathbf{h}^{(n)}$ instead of $\mathbf{z}^{(n)}$, where $\mathbf{h}^{(n)}$ corresponds to the output of the backbone convolutional network for the nth sample and $\mathbf{z}^{(n)}$ to the output of the embedding layer. Table 6 shows the benefits of diversely supervised search in the embedding space. When relying on a backbone model without fine-tuning, we obtain very low scores when trained either in supervised or self-supervised settings on ImageNet [54]. For the self-supervised model, we rely on MoCo v2 [10, 20]. When fine-tuning the model on the respective datasets, the diversely supervised search improves considerably. Searching in the embedding space is the most effective, as it captures all label similarities, and also the most efficient, as the dimension is lower than the feature space. For example in Diverse-Cars, the dimensionality of the embedding space is 200 compared with 2,048 in the feature space. When swapping backbones and searching in the feature space, we observe a similar behavior as in Table 5 where the performance drops close to zero. Diversely supervised search benefits from a retrieval operation in an embedding space that captures all label types.

Influence of diverse labels. We investigate the influence of each diverse label as a supervision source during training in Figure 4. In particular, we evaluate the effect of the instance, category, and all attributes labels individually and their combination. When leveraging all types of labels, it achieves the best overall scores. In general, the instance labels always matters and combining two types of labels leads to an improvement. Though, both product datasets exhibit different behaviors. Figure 4(a) shows that the model benefits the most from category labels on Diverse–Cars. Category labels alone yield a high retrieval score and combining them with other types of labels results in even higher scores. Indeed, car makers usually distill a similar design to all their car models. Being able to represent categories is then the most important. Figure 4(b) rather depicts the importance of attribute-label supervision on Diverse–Clothes. Attribute labels alone yield a high retrieval score and their combination with other labels is always beneficial. Contrary to car makers, fashion designers focus more on attribute combinations to create new products. Indeed, compared to cars, clothes have more attributes, which makes this supervision the most important.

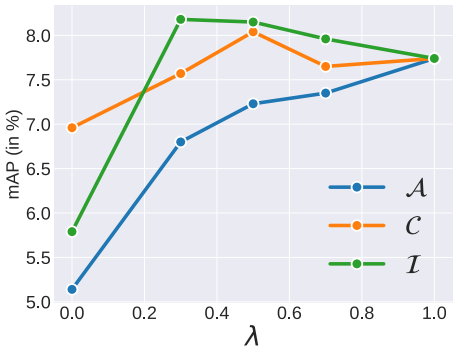


Fig. 5. Weighted diverse labels effect on the diverse search of all queries (mAP) on Diverse–Clothes. Performance can be slightly improved by reducing the contribution of the attributes or the category labels. For simplicity, we set all contributions to one.

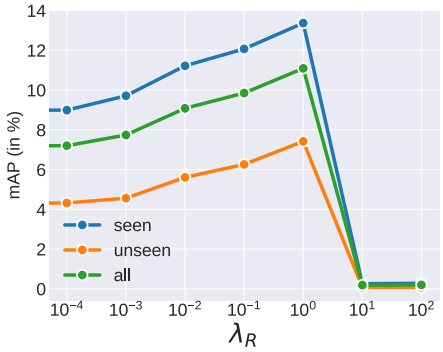


Fig. 6. Embedding regularization on Diverse–Clothes. The stronger the regularization, the higher the performance on diverse search is for both seen and unseen queries (mAP). Though, when the regularization is too strong, the model does not learn properly, as everything is pushed to zero.

In case of scarce resources, we then recommend to collect annotations on instances and categories for cars, and on instances and attributes for clothes.

Weighted diverse labels. While Figure 4 switches on and off the contribution of every lambda, Figure 5 evaluates these tradeoff hyper-parameters with real values. When evaluating every lambda individually, we fix the others to one. All settings improve over the absence of a label, which indicates that all labels are important to diversely supervised search. It is possible to slightly improve the performance by reducing the contribution of attributes or category labels on Diverse–Clothes rather than setting them all to one. Though, as the search space for the lambda triplet is vast, we recommend to simply set all three to one. Notably, this enables a simple, non-exhaustive, and fair comparison with alternative methods.

Embedding regularization. Figure 6 varies the amount of regularization λ_R on the embedding space on Diverse–Clothes. The higher the regularization, the better the performance of diversely supervised search is. Interestingly, this affects both seen and unseen queries positively. There is a cliff in performance after λ_R=1 where the performance drops drastically. Indeed, when the regularization is too strong, the representation is pushed towards zero, which annihilates the model learning.

Improving the performance. Comparisons in Table 1 are done with $\lambda_R = 0.001$. As shown in Figure 6, increasing this value can greatly benefit the diversely supervised search in our proposed models. Indeed, when applying a $\lambda_R = 1$ during training, we improve the mAP for all composite queries to 34.24 ± 0.23 for Diverse–Cars and to 11.34 ± 0.21 for Diverse–Clothes. Though, applying such a high regularization for the alternatives can be detrimental. For example, on Diverse–Cars, modulation drops to an mAP below one while conditional drops by five points. The fact that our model incorporates a grouping mechanism helps to benefit from higher regularization on the embedding space, as alternatives without any grouping suffer to various extents.

Influence of the number of attributes. We examine the influence of the number of attributes in the composite queries on the retrieval performance. As described in Section 4.2, we create composite queries with up to three attributes. For example, "a *DS* with *3* doors and *5* seats" is a composite

Table 7. Influence of the Number of Attributes

(a) **Diverse–Cars**

#A	seen	unseen	all
1	42.15	23.50	37.33
2	36.76	20.67	30.89
3	30.91	17.41	24.93

(b) **Diverse–Clothes**

#A	seen	unseen	all
1	20.44	4.29	18.79
2	10.05	4.40	8.53
3	7.86	4.61	6.34

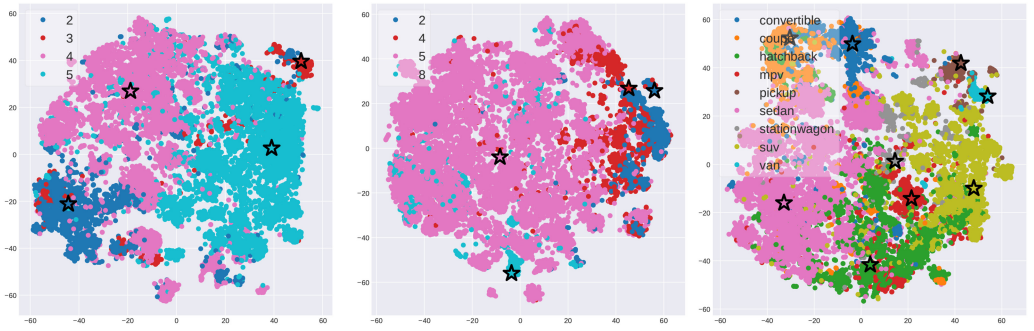
We examine the influence of the number of attributes in composite queries and report the mAP (in %). The more specific the composite query is, the harder it gets to retrieve relevant images. Unseen queries for clothes remain at the same level, because they are equally challenging, as the median number of images per query is the same.

(a) **Diverse–Cars** Our model can retrieve the multiple (third row) or only (second) matching car models. Yet, it can be fooled by cars of the same color (first).

(b) **Diverse–Clothes** Our model can retrieve rare (first row) or original (second and third) clothes items.

Fig. 7. Influence of the number of attributes. We show examples of *unseen* composite queries with an increasing number of attribute values and their top-five retrieved images (correct in *green*, incorrect in *red*).

query with a category and two attributes, for a total of three labels. Table 7 shows that increasing the number of attributes in the composite queries leads to a more challenging task. The more specific the search is, the harder it gets to find the needle in the haystack. On both datasets there is a drop of about 12 mAP when switching from one to three attributes. In particular, Table 7(a) exhibits a drop of only 6 mAP points for unseen queries but 11 mAP points for seen queries on Diverse–Cars. Table 7(b) shows a constant performance for unseen queries, while scores decrease more importantly for seen queries on Diverse–Clothes. This is explained by the fact that the median number of images per unseen composite query for all levels of detail is the same, making them equally challenging. Figure 7 depicts composite query examples with an increasing number of attributes. For Diverse–Cars, there can exist multiple car models matching the query. The model can retrieve correct images regardless of the viewpoint. Yet, confusion can happen when cars are of the same color or shape. For Diverse–Clothes, the search is more challenging, as there is usually one clothes item with very few images to retrieve. Items can be rare or exhibit original combinations of labels. Future work on product search should emphasize the retrieval performance of (a) composite queries with several attributes, as distinguishing products on a fine-grained level requires a higher amount of attributes, and (b) unseen composite queries as designers usually create products with an unseen combination of labels.



(a) **Number of doors** (b) **Number of seats** (c) **Type**

Fig. 8. Attribute subspace visualization with t-SNE on the test set of Diverse–Cars. Learned prototype representations for every value of every attribute are illustrated with a star. (a) The number of doors are clustered with prototypes being at extremities. Cars with three doors tend to spread all across the embedding space. (b) A transition from two seats to more than five seats is observed. Cars with more than five seats tend to spread all across the embedding space. (c) Car types are occupying the whole embedding space. Certain car types tend to be close to each other, e.g., *coupe* and *convertible*, or *pickup* and *van*.

Table 8. Binary Representations on the Diverse Search of All Queries (mAP) on Diverse–Cars and Diverse–Clothes

Representation	Cars	Clothes
Binary	21.47	7.58
Real-valued	**32.19**	**7.74**

While a binary representation has a large gap to real-valued representations on Diverse–Cars, it provides a compelling alternative with a close score on Diverse–Clothes.

Table 9. Sentence Representations on the Diverse Search of All Queries (mAP) on Diverse–Cars and Diverse–Clothes

Representation	Cars	Clothes
Sentences	5.51	3.96
Subspaces	**32.19**	**7.74**

Sentences cannot capture the diversity of all labels in composite queries, as they lack the flexibility of subspaces to represent every label.

Attribute subspace visualization. Figure 8 plots the t-SNE [42] visualization of every attribute subspace on the test set of Diverse–Cars, as well as the latent prototype visualization for every attribute value. For the number of doors attribute, the prototypes are well separated, with a prototype at each extremity. Though, it appears that cars with three doors do not have a compact representation, as they tend to spread all across the space. For the number of seats attribute, there is a transition from two seats to cars with more than five seats. This indicates that the model has found a progressive way to represent this attribute. For the type attribute, every car type is also represented around the region of its corresponding latent prototype. Some values are close to each other; for example, coupe and convertible, which indicates that the model has captured the car shape similarities.

Binary representations. As we design our embedding model to be a probabilistic model, a binary representation can also be used for diversely supervised search. In this scenario, the representation of every image corresponds to the one-hot predictions of the probabilistic model for the category label and every attribute label. Similarly, the composite query is represented by a one-hot binary representation for diversely supervised search. Table 8 compares the one-hot binary representations with real-valued representations. On Diverse–Cars, there is a large gap in performance between both representations. This difference resides in the fact that the performance for unseen queries drops by a factor two. On Diverse–Clothes, the performance is similar for both

(a) Typical instances (b) Atypical instances (c) Eclectic instances

Fig. 9. Discovering typical, atypical, and eclectic products. We explore product instances in the gallery set to discover design styles. Images in the same row share the same category label (underlined). The blue text box indicates the model prediction (*italics*). (a) Typical instances close to the category prototype depict the common appearance of sweaters, dresses, and shirts. (b) Atypical instances far from the category prototype exhibit a global appearance that resembles other categories, which causes misclassification. (c) Eclectic instances with a high entropy display original attribute values for the category.

representations, which suggests that binary representations can be used if storage space becomes a challenge.

Text representations. An alternative to learned label subspaces is to rely on text representations. The idea is to process the diverse label through a language model to obtain a text representation. The model then learns to regress to the text embedding, which is considered as a prototype during learning. For example, for an image with a diverse label "a shirt with long sleeves and a stripe print," we feed this sentence to a language model and take the output embedding as a prototype to regress to. We rely on sentence-BERT [51], a variant of BERT [13] for sentences fine-tuned on natural language inference datasets [7, 70], to extract text representations. Table 9 shows that text representations underperform learned label subspaces. Recall the example above. While the image has a diverse label "a shirt with long sleeves and a stripe print," it should hit for composite queries such as "a shirt with long sleeves" or "a shirt with a stripe print." Having a sentence representation is too rigid, as it imposes an order in the attributes and ties strongly attributes with the category. Instead, subspaces offer a more flexible representation and allow composite queries with various numbers of attributes in an unordered manner.

5.3 Discovering Typical, Atypical, and Eclectic Products

In this experiment, we aim to discover products with *typical*, *atypical*, or *eclectic* styles in the gallery set. We rely on the *triple* grouping that integrates attribute, category, and instance representations within the same embedding space. First, we aggregate visual representations per instance, i.e., images of the same instances are aggregated to the same visual representation. We refer to those as product representations. Second, we compute distances between product representations and all category prototypes $\mathbf{c}_y \in \mathbb{Z}_C$ in the embedding space. These distances provide three different indicators: (a) a small distance to the corresponding prototype indicates *typical* products, while (b) a large distance refers to *atypical* products. Additionally, the entropy can be computed over the probability distributions for each product representation, where (c) a high entropy refers to *eclectic* products on the edge of several categories. Probabilities are obtained by applying the softmax function over the distances.

We provide qualitative results based on the three indicators on Diverse–Clothes. Figure 9(a) illustrates the closest instances to category prototypes. These instances depict a common style, which makes them easily recognizable, as they form typical instances [52]. Distilling a typical design style in instances is particularly attractive for brands to enforce loyalty or attachment [63]. Yet, product design styles have a determined lifespan [61], and other combinations of visual attributes defining the style will emerge next due to cyclic [4] or punctual [43] trends. Figure 9(b) illustrates the farthest instances to category prototypes. The global shape of these instances, either in size or fabric, makes them look like they are part of another category. For example, instances of "dresses" in row 2 look like "tees." Thus, the model can misclassify these instances. Figure 9(c) illustrates instances that confuse the embedding the most, as they exhibit a high entropy. These instances depict an original visual appearance, especially for the *print* attribute. Searching for atypical and eclectic products reveals unexpected and intriguing trends in product design.

6 CONCLUSION

We have introduced the problem of diversely supervised visual product search, where queries describe a specific set of diverse labels to search for. We have proposed a diversely supervised embedding, where attribute, instance, and attribute labels provide a diverse supervision to learn a representation for products. Evaluation relies on composite queries to describe the specific set of labels to search for. Composite query representations correspond to a per-label average of selected visual representations in the embedding space. Experiments on seen and unseen settings show that our diversely supervised embedding better models a diverse set of labels than alternative baselines repurposed for diversely supervised visual product search. The embedding also enables the discovery of the typicality effect in design styles, which reveals intriguing products. In the current form, labels describe physical properties of products but could also capture aesthetics or cultural differences.

APPENDICES

In this Appendix, we present the labeling process for Diverse–Cars (Section A) and Diverse–Clothes (Section B).

A DIVERSE–CARS LABELING PROCESS

Diverse–Cars builds upon Cars196 by Krause et al. [33] and CompCars by Yang et al. [72]. Both datasets provide a large number of car models. Every car model comprises multiple images from several viewpoints. CompCars [72] already comes with an initial set of labels, while Cars196 [33] has only car model labels. By merging both datasets, Diverse–Cars covers car models sold in both North American and South-Pacific regions.

We manually re-annotate the images to ensure the quality of the *category* and *attribute* labels. Besides the new category and attribute labels, we also ensure that similar car models between the two original datasets are merged. The new labels will be made public.

In the newly proposed labels, *category* and *attribute value* labels are annotated. Original *instance* labels are preserved. We adopt the same three attribute vocabularies as initially defined in CompCars [72]. Figure 10 shows one sample for every attribute value of every attribute.

Overall, a total of 691 unique instances are annotated. Every image in the dataset receives an instance, a category, and three attribute value labels. Note that some categories are very scarce. We ensure that there are at least one or two models per car maker in the training set, which in return can result in the absence of some car makers in the gallery set. In other words, not all car makers are present in the gallery set.

For hyper-parameters search, we create a separate validation set from the training set. We randomly sample 17 car models, for a total of 1,169 images. We keep the validation separate. There is no re-training on both training and validation sets once hyper-parameters are fixed.

B DIVERSE–CLOTHES LABELING PROCESS

Diverse–Clothes builds upon In-Shop Clothes by Liu et al. [40], which provides a large number of clothing products. Every product comprises multiple images from several viewpoints and a rich description of several sentences. However, the labeling of the original In-Shop Clothes dataset was done in a weakly supervised manner, which can result in scarce attribute values, duplicates, or incoherencies [75].

We manually re-annotate the images to ensure the quality of the *category* and *attribute* labels. Besides the new category and attribute labels, other cleaning tasks are also performed: (1) instance and image duplicates are removed; (2) instances with two different category labels are merged. The new labels will be made public.

In the newly proposed labels, *category* and *attribute value* labels are re-annotated. Original *instance* labels are preserved. Eight different new *attributes* are defined. Figure 11 shows one sample for every value of every attribute.

Overall, a total of 7,978 unique instances are re-annotated. While a category label and an instance label are assigned to all instances, not all attribute labels are necessarily assigned to all instances. For example, a *skirt* does not have a *sleeves length* attribute label.

For hyper-parameters search, we create a separate validation set from the training set. We sample 59 clothes items for a total of 352 images. We keep the validation separate and do not re-train on it once hyper-parameters are fixed.

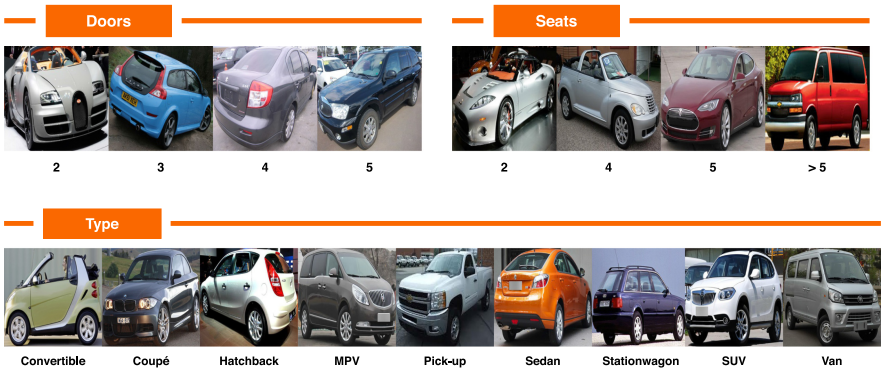


Fig. 10. Image samples for every attribute value in Diverse–Cars.

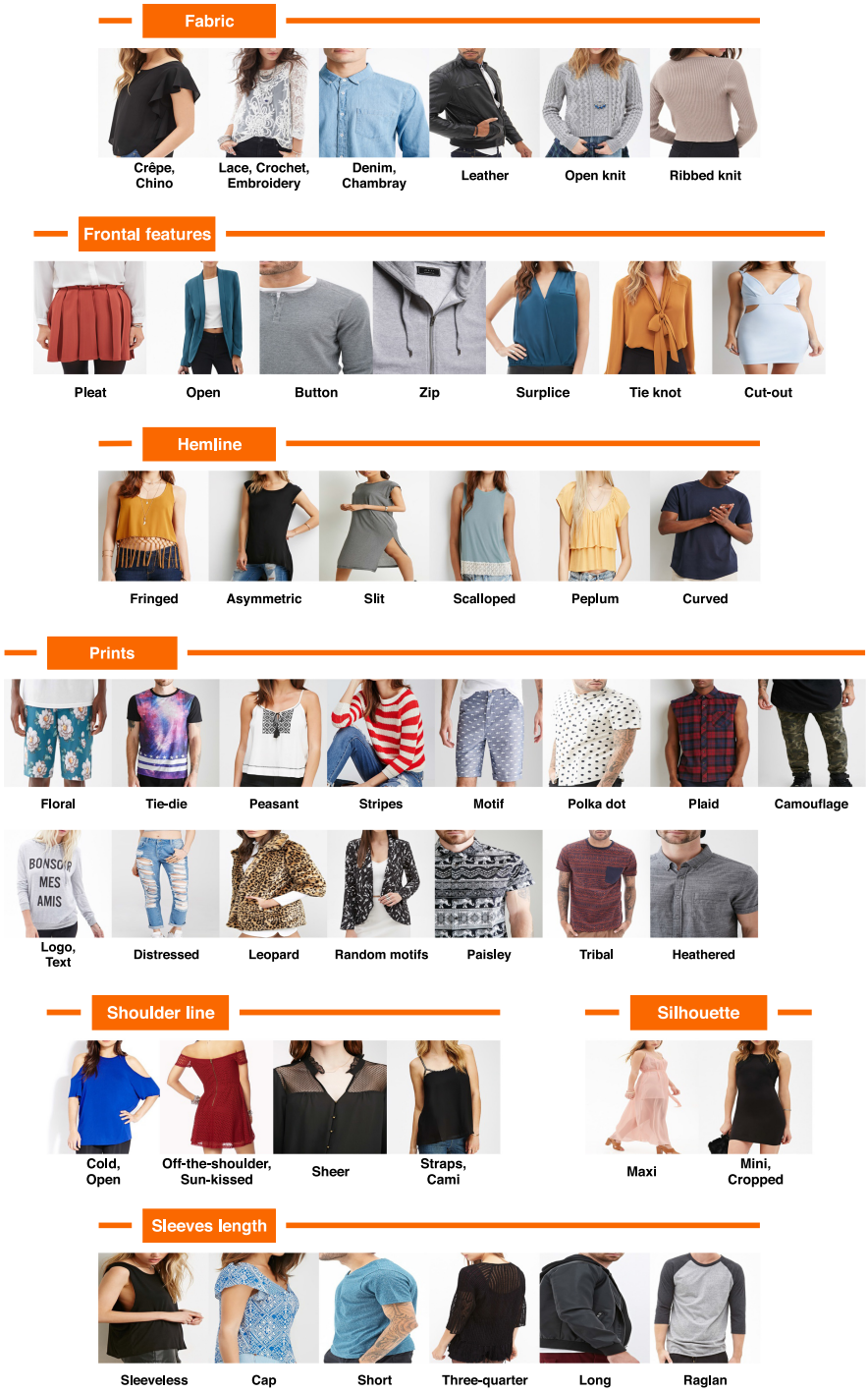


Fig. 11. Image samples for every attribute value in Diverse–Clothes.

REFERENCES

[1] Kenan E. Ak, Ashraf A. Kassim, Joo Hwee Lim, and Jo Yew Tham. 2018. Learning attribute representations with localization for flexible fashion search. In *CVPR*.

[2] Kenan E. Ak, Joo Hwee Lim, Jo Yew Tham, and Ashraf A. Kassim. 2019. Attribute manipulation generative adversarial networks for fashion images. In *ICCV*.

[3] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2016. Label-embedding for image classification. *IEEE Trans. Pattern Anal. Mach. Intell.* 38, 7 (2016).

[4] Ziad Al-Halah, Rainer Stiefelhagen, and Kristen Grauman. 2017. Fashion forward: Forecasting visual style in fashion. In *ICCV*.

[5] Sean Bell and Kavita Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.* 34, 4 (2015).

[6] Alessandro Bergamo, Lorenzo Torresani, and Andrew W. Fitzgibbon. 2011. PiCoDes: Learning a compact code for novel-category recognition. In *NeurIPS*.

[7] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

[8] Chandramani Chaudhary, Poonam Goyal, Navneet Goyal, and Yi-Ping Phoebe Chen. 2020. Image retrieval for complex queries using knowledge embedding. *ACM Trans. Multim. Comput., Commun. Applic.* 16, 1 (2020).

[9] Gal Chechik, Uri Shalit, Varun Sharma, and Samy Bengio. 2009. An online algorithm for large scale image similarity learning. In *NeurIPS*.

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020. Improved baselines with momentum contrastive learning. In *arXiv:2003.04297*.

[11] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*.

[12] Thomas Deselaers and Vittorio Ferrari. 2011. Visual and semantic similarity in ImageNet. In *CVPR*.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

[14] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing objects by their attributes. In *CVPR*.

[15] Vittorio Ferrari and Andrew Zisserman. 2008. Learning visual attributes. In *NeurIPS*.

[16] Andrea Frome, Yoram Singer, and Jitendra Malik. 2007. Image retrieval and classification using local distance functions. In *NeurIPS*.

[17] Albert Gordo and Diane Larlus. 2017. Beyond instance-level image retrieval: Leveraging captions to learn a global visual representation for semantic retrieval. In *CVPR*.

[18] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *CVPR*.

[19] Xintong Han, Zuxuan Wu, Phoenix X. Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S. Davis. 2017. Automatic spatially aware fashion concept discovery. In *ICCV*.

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

[22] Wei-Lin Hsiao and Kristen Grauman. 2018. Creating capsule wardrobes from fashion images. In *CVPR*.

[23] Junshi Huang, Rogerio S. Feris, Qiang Chen, and Shuicheng Yan. 2015. Cross-domain image retrieval with a dual attribute-aware ranking network. In *CVPR*.

[24] Junshi Huang, Si Liu, Junliang Xing, Tao Mei, and Shuicheng Yan. 2014. Circle & search: Attribute-aware shoe retrieval. *ACM Trans. Multim. Comput., Commun. Applic.* 11, 1 (2014).

[25] Yushi Jing, David Liu, Dmitry Kislyuk, Andrew Zhai, Jiajing Xu, Jeff Donahue, and Sarah Tavel. 2015. Visual search at Pinterest. In *KDD*.

[26] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. 2015. Image retrieval using scene graphs. In *CVPR*.

[27] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

[28] 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. In *ICCV*.

[29] M. Hadi Kiapour, Kota Yamaguchi, Alexander C. Berg, and Tamara L. Berg. 2014. Hipster wars: Discovering elements of fashion styles. In *ECCV*.

[30] Gunhee Kim, Seungwhan Moon, and Leonid Sigal. 2015. Ranking and retrieval of image sequences from multiple paragraph queries. In *CVPR*.

[31] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

[32] Adriana Kovashka, Devi Parikh, and Kristen Grauman. 2012. WhittleSearch: Image search with relative attribute feedback. In *CVPR*.
[33] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3D object representations for fine-grained categorization. In *ICCVw*.
[34] Neeraj Kumar, Alexander Berg, Peter N. Belhumeur, and Shree Nayar. 2011. Describable visual attributes for face verification and image search. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 10 (2011).
[35] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. 2014. Attribute-based classification for zero-shot visual object categorization. *IEEE Trans. Pattern Anal. Mach. Intell.* 36, 3 (2014).
[36] Tian Lan, Weilong Yang, Yang Wang, and Greg Mori. 2012. Image retrieval with structured object queries using latent ranking SVM. In *ECCV*.
[37] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. 2018. Stacked cross attention for image-text matching. In *ECCV*.
[38] Si Liu, Zheng Song, Guangcan Liu, Changsheng Xu, Hanqing Lu, and Shuicheng Yan. 2012. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *CVPR*.
[39] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*.
[40] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. 2016. DeepFashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*.
[41] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic gradient descent with warm restarts. In *ICLR*.
[42] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *J. Mach. Learn. Res.* 9, Nov. (2008).
[43] Utkarsh Mall, Kevin Matzen, Bharath Hariharan, Noah Snavely, and Kavita Bala. 2019. GeoStyle: Discovering fashion trends and events. In *ICCV*.
[44] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY.
[45] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *ACM SIGIR*.
[46] Yair Movshovitz-Attias, Alexander Toshev, Thomas K. Leung, Sergey Ioffe, and Saurabh Singh. 2017. No fuss distance metric learning using proxies. In *ICCV*.
[47] Devi Parikh and Kristen Grauman. 2011. Relative attributes. In *ICCV*.
[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*.
[49] Mohammad Rastegari, Ali Diba, Devi Parikh, and Ali Farhadi. 2013. Multi-attribute queries: To merge or not to merge? In *CVPR*.
[50] Hareesh Ravi, Lezi Wang, Carlos Muniz, Leonid Sigal, Dimitris Metaxas, and Mubbasir Kapadia. 2018. Show me a story: Towards coherent neural story illustration. In *CVPR*.
[51] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *EMNLP*.
[52] E. H. Rosch. 1978. Principles of categorization. *Cognition and Categorization*, Lawrence Erlbaum (Ed.) (1978).
[53] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019. Latent multi-task architecture learning. In *AAAI*, Vol. 33.
[54] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* 115, 3 (2015).
[55] Mohammad Amin Sadeghi and Ali Farhadi. 2011. Recognition using visual phrases. In *CVPR*.
[56] Walter J. Scheirer, Neeraj Kumar, Peter N. Belhumeur, and Terrance E. Boult. 2012. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *CVPR*.
[57] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *CVPR*.
[58] Behjat Siddiquie, Rogerio S. Feris, and Larry S. Davis. 2011. Image ranking and retrieval based on multi-attribute queries. In *CVPR*.
[59] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.
[60] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. 2016. Deep metric learning via lifted structured feature embedding. In *CVPR*.
[61] George B. Sproles. 1981. Analyzing fashion life cycles—Principles and perspectives. *J. Market.* 45, 4 (1981), 116–124.
[62] Gjorgji Strezoski, Nanne van Noord, and Marcel Worring. 2019. Many task learning with task routing. In *ICCV*.

[63] Douwe Van den Brink, Gaby Odekerken-Schröder, and Pieter Pauwels. 2006. The effect of strategic and tactical cause-related marketing on consumers' brand loyalty. *J. Consum. Market.* 23, 1 (2006).
[64] Andreas Veit, Serge Belongie, and Theofanis Karaletsos. 2017. Conditional similarity networks. In *CVPR*.
[65] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *ICCV*.
[66] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li-Jia Li, Li Fei-Fei, and James Hays. 2019. Composing text and image for image retrieval—An empirical odyssey. In *CVPR*.
[67] Yu-Xiong Wang and Martial Hebert. 2016. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*.
[68] Zihao Wang, Xihui Liu, Hongsheng Li, Lu Sheng, Junjie Yan, Xiaogang Wang, and Jing Shao. 2019. Camp: Cross-modal adaptive message passing for text-image retrieval. In *ICCV*.
[69] Kilian Q. Weinberger and Lawrence K. Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.* 10 (2009).
[70] Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.
[71] Fan Yang, Ajinkya Kale, Yury Bubnov, Leon Stein, Qiaosong Wang, Hadi Kiapour, and Robinson Piramuthu. 2017. Visual search at eBay. In *KDD*.
[72] L. Yang, P. Luo, C. C. Loy, and X. Tang. 2015. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*.
[73] Linwei Ye, Zhi Liu, and Yang Wang. 2018. Learning semantic segmentation with diverse supervision. In *WACV*.
[74] A. Yu and K. Grauman. 2014. Fine-grained visual comparisons with local learning. In *CVPR*.
[75] R. Zakizadeh, M. Sasdelli, Y. Qian, and E. Vazquez. 2018. Improving the annotation of DeepFashion images for fine-grained attribute recognition. *arXiv:1807.11674* (2018).
[76] Andrew Zhai, Dmitry Kislyuk, Yushi Jing, Michael Feng, Eric Tzeng, Jeff Donahue, Yue Li Du, and Trevor Darrell. 2017. Visual discovery at Pinterest. In *WWW*.
[77] Andrew Zhai and Hao-Yu Wu. 2019. Classification is a strong baseline for deep metric learning. In *BMVC*.
[78] Andrew Zhai, Hao-Yu Wu, Eric Tzeng, Dong Huk Park, and Charles Rosenberg. 2019. Learning a unified embedding for visual search at Pinterest. In *KDD*.
[79] Yanhao Zhang, Pan Pan, Yun Zheng, Kang Zhao, Yingya Zhang, Xiaofeng Ren, and Rong Jin. 2018. Visual search at Alibaba. In *KDD*.
[80] Bo Zhao, Jiashi Feng, Xiao Wu, and Shuicheng Yan. 2017. Memory-augmented attribute manipulation networks for interactive fashion search. In *CVPR*.
[81] Xiangyun Zhao, Haoxiang Li, Xiaohui Shen, Xiaodan Liang, and Ying Wu. 2018. A modulation module for multi-task learning with applications in image retrieval. In *ECCV*.
[82] Shizhan Zhu, Raquel Urtasun, Sanja Fidler, Dahua Lin, and Chen Change Loy. 2017. Be your own Prada: Fashion synthesis with structural coherence. In *ICCV*.

Received July 2020; revised April 2021; accepted April 2021