Computer Vision and Image Understanding 182 (2019) 81–92

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Domain invariant hierarchical embedding for grocery products recognition

Alessio Tonioni *, Luigi Di Stefano

University of Bologna, Department of Informatic Science and Engineering, Viale del risorgimento 2, Bologna, Italy

ARTICLE INFO

Communicated by N. Paragios

Keywords:
Grocery products recognition
Embedding learning
Generative adversarial networks
Object recognition

ABSTRACT

Recognizing packaged grocery products based solely on appearance is still an open issue for modern computer vision systems due to peculiar challenges. Firstly, the number of different items to be recognized is huge (i.e., in the order of thousands) and rapidly changing over time. Moreover, there exist a significant domain shift between the images that should be recognized at test time, taken in stores by cheap cameras, and those available for training, usually just one or a few studio-quality images per product. We propose an end-to-end architecture comprising a GAN to address the domain shift at training time and a deep CNN trained on the samples generated by the GAN to learn an embedding of product images that enforces a hierarchy between product categories. At test time, we perform recognition by means of K-NN search against a database consisting of just one reference image per product. Experiments addressing recognition of products present in the training datasets as well as different ones unseen at training time show that our approach compares favorably to state-of-the-art methods on the grocery recognition task and generalize fairly well to similar ones.

1. Introduction

Automatic recognition of grocery products is receiving increasing attention as it may lead to improved shopping experience (e.g., shopping apps, interaction via augmented reality, checkout-free stores, support to visually impaired customers) as well as to a more efficient store management (e.g., by automated inventory and on-line shelf monitoring).

As pointed out by Merler et al. (2007) in their seminal work, recognizing grocery products can be thought of as an object recognition problem featuring peculiar challenges. Firstly, the number of different items to be recognized is huge, in the order of several thousands for small to medium shops, well beyond the usual target for current state-of-the-art image classifiers. Furthermore, product recognition is better cast as an hard instance recognition rather than a classification problem, as it mandates telling apart many items looking identical but for a few details, such as the different flavors of the same cereal brand depicted in Figs. 1(b–d). Any practical methodology should rely only on the model images available within existing commercial product databases, i.e. a single high-quality image for each side of the package either acquired in studio settings or rendered using computer graphics tools (e.g., Figs. 1(c–d)). Conversely, products must be recognized from images captured in the store by cheap sensors, e.g., smartphone cameras, under far less than ideal conditions (e.g., Fig. 1a). Thus, product recognition implies tackling a domain adaptation problem between the images available to build the inference engine and those deployed at test time to pursue recognition. Another peculiarity deals with the items on sale in a store as well as their appearance changing frequently overtime, which would render unfeasible to continuously re-train or fine-tune an inference engine in order to keep-up with all such changes. Differently, a practical approach should be conducive to recognition of both *seen products* (i.e., products whose reference images were deployed at training time) as well as *unseen products* (i.e., products present in the store but not used to train the inference engine).

Given these premises, we rely on a global image descriptor learned to disentangle grocery products and to pursue recognition through K-NN search within a database featuring one reference image per sought product. This approach allows for learning an image embedding based on the available training data and then perform recognition of both *seen* and *unseen* products seamlessly. For example, should a new product be put on sale in the store, our system would just require to add one image of the new product into the reference database without the need of performing a new costly retraining. K-NN search is quite amenable to product recognition from a computational standpoint alike. Indeed, compared to typical image retrieval settings, the database is very small (i.e., in the order of several thousands images rather than millions of images). Thus, a global image descriptor of about a few hundreds entries turns out viable in terms of time and memory efficiency.

We propose to learn the embedding through a deep CNN trained by a loss function that forces both similar looking items as well as items belonging to the same high level category to map close one to another in the descriptor space. Moreover, to tackle the domain shift issue and increase the training set size we propose to deploy an image-to-image translation GAN together with the embedding CNN and to

* Corresponding author.
E-mail address: alessio.tonioni@unibo.it (A. Tonioni).

https://doi.org/10.1016/j.cviu.2019.03.005
Received 11 July 2018; Received in revised form 9 January 2019; Accepted 23 March 2019
Available online 27 March 2019
1077-3142/

(a) (b) (c) (d)

Fig. 1. Exemplar images for the grocery recognition task: *reference* images (c–d) carefully acquired in studio (available at training time), *query* images: (a) captured in the store (*query* at test time) and (b) *synthetic query* generated by our GAN (used at training time).

optimize the whole architecture end-to-end. In particular, some of the training samples for the embedding network are generated by a GAN that learns unsupervisedly to transform images taken in studio settings into in-store type of images without introducing excessive modification of product appearance (Fig. 1(b) shows an exemplar image synthesized by the GAN). These training samples force the embedding CNN to learn robustness to domain shift; moreover, the GAN can be trained to produce samples that are particularly hard to embed, thereby allowing the CNN to learn a stronger embedding function thanks to these adversarial samples. Despite the use of multiple networks, the overall architecture can be trained end-to-end effortlessly via simple gradient descent.

Thus, the main original contributions of this paper can be summarized as follows.

- The introduction of an image-to-image translation GAN trained jointly together with an embedding network in order to produce a domain-shift resilient and stronger embedding function. To the best of our knowledge, this is the first work that tries to integrate a GAN with adversarial behavior within the training process of an embedding network.
- A novel formulation of the classic triplet ranking loss that can be used to learn a better embedding whenever in the domain of interest there exist a taxonomy between classes (e.g., ImageNet class taxonomy). Our loss helps preserving in the descriptor space the similarity information implied by the taxonomy (i.e., items belonging to the same macro class should be embedded closer than those belonging to unrelated classes).
- A novel formulation of the grocery product recognition task as an instance-level recognition problem with thousands of classes and only one sample per class. In Section 4 we will show how in this real-world scenario, far more challenging than the standard datasets currently used to benchmark methods aimed at learning from few shots, our proposed architecture can obtain impressive performance.

2. Related work

Grocery product recognition. The grocery products recognition problem was firstly investigated in the seminal paper by Merler et al. (2007). Together with a thoughtful analysis of the problem, the authors propose a system based on local invariant features to help visually impaired costumers shop in grocery stores. However, their experiments report performance far from conducive to real-world deployment. A number of more recent proposals are aimed at improving product recognition by leveraging on: (a) stronger features followed by classification (Cotter et al., 2014), (b) the statistical correlation between nearby products on shelves (Advani et al., 2015; Baz et al., 2016) (c) information on the expected product disposition (Tonioni and Di Stefano, 2017) or (d) a hierarchical multi-stage recognition pipeline (Franco et al., 2017). Yet, all these recent works focus on a relatively small-scale problem, i.e. recognition of a few hundreds different items at most, whilst several thousands products are on sale in a real shop. George and Floerkemeier (2014) address more realistic settings and propose a quite complex multi-stage system capable of recognizing $\sim$ 3400 different products based on a single model image per product. The authors contribution include releasing the dataset employed in their experiments, which we will use in our evaluation. Recently Karlinsky et al. (2017) have shown how it is possible to improve detection and recognition performance on the same dataset relying on a probabilistic model based on local feature matching and refinement by deep network. However, even in this work, performance appear not as satisfactory as to pave the way for practical exploitation.

Computer vision for retail. The recognition of grocery products shares commonalities with the *exact street to shop* task addressed in Hadi Kiapour et al. (2015), Wang et al. (2016), Shankar et al. (2017), which consists in recognizing a real-world example of a garment item based on the catalog of an online shop. Like our solution, these works rely on matching and retrieval using deep features extracted from CNN architectures. However, they leverage on labeled paired couple of samples depicting the same item in the *Street* and *Shop* domain to learn a cross domain embedding at training time while our proposal leverages only on labeled images from one domain, thereby vastly relaxing the applicability constraint. Moreover, computer vision has been successfully applied in the retail environments for costumer profiling (Sturari et al., 2016), automatic shelf surveying (Paolanti et al., 2017), visual market basket analysis (Santarcangelo et al., 2018) and automatic localization inside the store (Wang et al., 2015).

Embedding learning. Using CNNs to obtain rich image representations is nowadays an established approach to pursue image retrieval, both as a strong off-the-shelf baseline (Sharif Razavian et al., 2014) and as a key component within more complex pipelines (Gordo et al., 2016). Schroff et al. (2015) train a CNN using triplets of samples to create an embedding for face recognition and clustering. Since then, this approach has been used extensively to learn representations for a variety of different tasks, with more recent works advocating smart sampling strategies (Wu et al., 2017) or suitable regularizations (Zhang et al., 2017) to ameliorate performance. Similarly to our proposal, Zhang et al. (2016) extend the idea of triplets by a novel formulation amenable to embed label structure and semantics at training time based on tuplets. Unlike (Zhang et al., 2016; Schroff et al., 2015), in this paper we propose to embed label structure within the learning process using only standard triplets; moreover our method uses only one exemplar image per class and augment the training set by a GAN trained jointly together with the embedding network.

Few shot learning. Few shot learning has been addressed successfully by Hariharan and Girshick (2017) through classifiers trained on top of a fixed feature representation by artificially augmenting a small training set with transformations in the feature space. Yet, in the grocery product recognition scenario the items to be recognized at test time change quite frequently, which would mandate frequent retraining of new classifiers. Besides, as product packages exhibit very low intra-class variability, the generalization ability of a classifier may not be needed. Thus, we prefer to learn a strong image embedding and rely on K-NN similarly to perform recognition. In this respect our approach shares commonalities with Vinyals et al. (2016) and Snell et al. (2017), where the authors address few shot learning by learning suitable embedding spaces and matching functions.

GANs. Starting from the pioneering works of Goodfellow et al. (2014) and Radford et al. (2016), GANs have received ever increasing attention in the computer vision community as they enable to synthesize realistic images with few supervision. Recently GAN frameworks have been successfully deployed to accomplish image-to-image translation, with (Isola et al., 2017) and without (Zhu et al., 2017; Shrivastava et al., 2017) direct supervision, as well as to tackle domain shift issues by forcing a classifier to learn invariant features Tzeng et al. (2017). We draw inspiration from these works and deploy a GAN at training time to pursue domain adaptation as well as to improve the effectiveness of the learned embedding. A related idea is proposed in Qiu et al. (2017) though, unlike Qiu et al. (2017), (a) we explicitly deploy the GAN while learning the embedding to attain domain adaptation, (b) use only one sample per class and (c) train the GAN to produce realistic though hard to embed training samples, i.e. the *generator* of our GAN not only plays an adversarial game against the *discriminator* but also against the *encoder* network that learns the embedding.

3. Domain invariant hierarchical embedding

An overview of our Domain invariant hierarchical embedding (DIHE) is depicted in Fig. 2. We use a deep CNN (*encoder*) to learn an embedding function $E : \mathcal{I} \rightarrow \mathcal{D}$ that maps an input image $i \in \mathcal{I}$ to a k-dimensional descriptor $d^k \in \mathcal{D}$ amenable to pursue recognition through K-NN similarity search. During training we exploit, if available in the training dataset, a taxonomy of classes by means of a novel loss function that forces descriptors of different items to be closer if they share some portion of the taxonomy, distant otherwise. To learn a descriptor robust to the domain shift between train and test data, we use an image-to-image translation GAN, consisting of a *generator* and a *discriminator*, which augments the training set with samples similar to those belonging to the test domain while simultaneously producing hard examples for the embedding network. The three networks can be trained jointly by standard gradient descent in order to minimize the three loss functions described in the following sections.

3.1. Hierarchical embedding

An established approach to learn an effective image embedding consists in training a deep CNN according to the *triplet ranking loss* (Wang et al., 2014). In the original formulation, each training sample consists of three different images, referred to as *anchor* (i_a), *positive* (i_p) and *negative* (i_n). In a typical classification scenario these images would be chosen such that i_a and i_p depict the same class while i_n belongs to a different one. Given a distance function in the descriptor space, $d(\mathbf{X}, \mathbf{Y})$, with $X, Y \in \mathcal{D}$, and denoted as $E(i)$ the descriptor computed by the encoder network for image i, the loss to be minimized is defined as

$$\mathcal{L}_{enc} = max(0, d(E(i_a), E(i_p)) - d(E(i_a), E(i_n)) + \alpha) \quad (1)$$

with α a fixed margin to be enforced between the pair of distances.

We modify this formulation for domains that feature a hierarchical structure between classes (e.g., ImageNet classes taxonomy), so as to mimic this structure within the learned descriptor space. This is a quite common scenario for problems where a multi level classification is available. For instance, existing commercial databases of grocery products feature labels both at instance as well as at multiple category levels (e.g., for the product depicted in Fig. 1(c) we would have three different classification labels with increasing generality: Kellog's Special K Classic →Cereal→Food). Our aim is to force the network to embed images nearby in the descriptor space not only based on their appearance but also on higher level semantic cues, like those shared between items belonging to the same macro-class. We argue that doing so will help producing a stronger image descriptor and may provide better generalization to products whose reference images are *unseen* at training time.

Assuming a taxonomy of classes encoded in a tree like structure, we propose to impose a hierarchy in $\mathcal{D}$ by rendering α inversely proportional to the *amount of hierarchy* shared between the classes of i_a, i_p and i_n. Considering an image sample i in the training set, we denote with c its fine class (foil level in the taxonomy) and with $\mathcal{H}(i)$ a set of higher level classes (all the parent nodes in the class tree excluding the common root). Using this notation and defining the minimum and maximum margin, α_{min} and α_{max} respectively, our hierarchical margin, $\alpha \in [\alpha_{min}, \alpha_{max}]$, can be computed as:

$$\alpha = \alpha_{min} + \left(1 - \frac{|\mathcal{H}(i_a) \cap \mathcal{H}(i_n)|}{|\mathcal{H}(i_a)|}\right) \cdot (\alpha_{max} - \alpha_{min}) \quad (2)$$

where $|\cdot|$ is the cardinality operator for sets. Thus, if i_a and i_b share all the parent nodes $\alpha = \alpha_{min}$, whilst the margin is proportionally increased until completely disjoint fine classes will produce $\alpha = \alpha_{max}$.

3.2. Domain invariance

A common trait across many computer vision tasks is that easily available labeled training data (e.g., tagged images published on-line) are usually sampled from a different distribution than the actual test images. Thus, machine learning models directly trained on such samples, such as embedding networks, will typically perform poorly at test time due to domain shift issues. However, annotating samples from the test distribution, even if possible, is usually very expensive and time consuming. We propose to address this problem by dynamically transforming the appearance of the available labeled training images to make them look similar to samples from the unlabeled test images. This transformation is carried out by two CNNs, refereed to in Fig. 2 as *generator* and *discriminator*, realizing an image-to-image translation GAN which is trained end-to-end together with the embedding network (*encoder*).

Given two image domains $\mathcal{A}, \mathcal{B} \subset \mathcal{I}$ consisting of $i^{\mathcal{A}} \in \mathcal{A}$ and $i^{\mathcal{B}} \in \mathcal{B}$, the standard image-to-image GAN framework can be summarized as a *generator* network that tries to learns a generative function $G : \mathcal{A} \rightarrow \mathcal{B}$ by playing a two player min–max game against a *discriminator* network $D : \mathcal{I} \rightarrow \mathbb{R}$ that tries to classify examples either as real images from $\mathcal{B}$ or fake ones produced by G. In the following we will denote with $G(i^{\mathcal{A}}) \in \mathcal{I}$ the output of the *generator* network given the input image $i^{\mathcal{A}}$ and with $D(i) \in \mathbb{R}$ the output of the *discriminator* network for image i. To generate samples similar to the images from domain $\mathcal{B}$ without drastically changing the appearance of the input image i^{A}, we introduce an additional term in the generator loss function ($\mathcal{L}_{reg}$) that, similarly to the self regularization term deployed in Shrivastava et al. (2017), forces $G(i^{A})$ to be visually consistent with i^{A}. In our architecture, $\mathcal{A}$ is the training set while $\mathcal{B}$ is a small set of unlabeled images from the test data distribution.

Following the notation introduced in Section 3.1, during each training iteration of the whole architecture we sample one image $i^{\mathcal{B}} \in \mathcal{B}$ to train the *discriminator* and two from the other domain $i_p^{\mathcal{A}}, i_n^{\mathcal{A}} \in \mathcal{A}$ to train the *encoder* and *generator*. As mentioned in Section 3.1, the *encoder* needs triplets of samples to compute its loss, so we synthesize the missing image using the *generator* $i_a^{\mathcal{B}} = G(i_p^{\mathcal{A}})$. With this architecture the triplet used to calculate Eq. (1) consists of two images from domain $\mathcal{A}$ and one from the simulated domain $\mathcal{B}$, thereby mimicking the test conditions where the query images to be recognized will came from $\mathcal{B}$ and the reference images to perform K-NN similarity from $\mathcal{A}$.

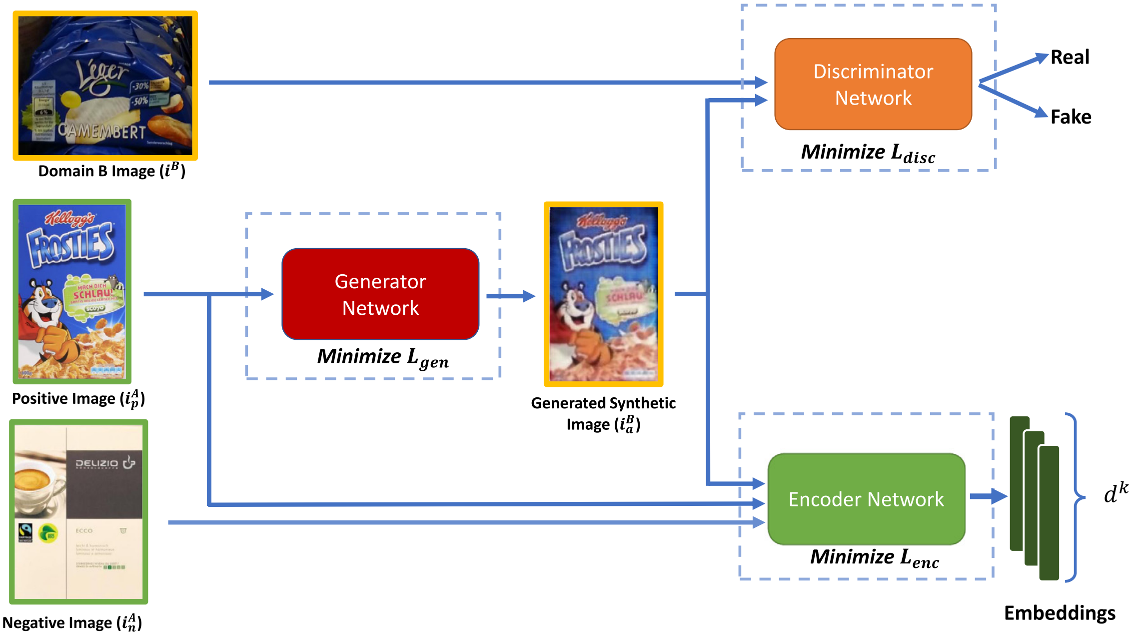


Fig. 2. Overview of DIHE at training time. Each training sample consists of three images, two from domain $\mathcal{A}$ (enclosed in green) and one from domain $\mathcal{B}$ (enclosed in yellow). The *generator* and *discriminator* implement a classic GAN for domain translation from $\mathcal{A}$ to $\mathcal{B}$. The *encoder* network uses two images from domain $\mathcal{A}$ alongside with the generated one to learn an image embedding by a modified triplet ranking loss. i_a^B is generated to be both indistinguishable from images sampled from domain $\mathcal{B}$ as well as hard to encode. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The *encoder* is trained to minimize Eq. (1) with the margin defined in Eq. (2). The *discriminator* tries to minimize a standard cross entropy loss:

$$\mathcal{L}_{disc} = log(D(i^B)) + log(1 - D(G(i_p^A))) \tag{3}$$

while the *generator* minimizes a loss consisting of three terms:

$$\begin{aligned} \mathcal{L}_{gen} &= L_{adv} + \lambda_{reg} \cdot L_{reg} + \lambda_{emb} \cdot L_{emb} \\ L_{adv} &= -log(D(G(i_p^A))) \\ L_{reg} &= \phi(i_p^A, i_a^B) \\ L_{emb} &= -d(E(i_p^A), E(G(i_p^A))) \end{aligned} \tag{4}$$

with $\phi(x, y)$, $x, y \in \mathcal{I}$ a similarity measure between the appearance of image x and y, either at pixel level (e.g., mean absolute difference…) or at image level (e.g., SAD, ZNCC…), and $\lambda_{reg}, \lambda_{emb}$ two hyper parameters that weigh the different terms of the loss function. We refer the reader to Section 4.1 for the actual $d(x, y,)$, $\lambda_{reg}, \lambda_{emb}$ and $\phi(x, y)$ used in the experiments. The contribution of the three terms can be summarized as follows. L_{adv} is the standard adversarial loss for the generator network that forces the synthesized images to be indistinguishable from those sampled from domain $\mathcal{B}$; L_{reg} is aimed at synthesizing images that preserve the overall structure of the input ones (avoiding thereby the mode collapse issue often occurring in GAN generators); L_{emb} forces an additional adversarial behavior against the *encoder*, so as to create hard to embed samples.

At training time, given a minibatch of M different triplets of samples (i_p^A, i_n^A, i^B), the three networks are trained jointly to minimize their average loss on the M samples.

4. Experimental results

4.1. Implementation details

Datasets. To evaluate the effectiveness of DIHE in recognizing grocery products we rely on two products datasets comprising thousands of items: the publicly available *Grocery Products* dataset (George and Floerkemeier, 2014) and a standard commercial database, referred to here as *Product8600*. Both datasets include more than 8500 grocery

Fig. 3. Visualization of the hierarchy of categories of the *Grocery_Food* dataset used as training set throughout our experiments. Each outermost category contains several different fine classes (products) not depicted for clarity.

products, each described by exactly one studio-quality (*reference*) image of the frontal face of the package, and feature a multi level class hierarchy in the categorization of products. As already discussed, at test time we pursue recognition from a different set of images (*query*). To create this set, for *Grocery Products* we automatically cropped individual items from the available shelf images according to the annotation released in Tonioni and Di Stefano (2017), thereby obtaining a total of 938 *query* images. As for *Product8600*, we cropped and annotated individual items from shelf videos that we acquired in a grocery store by a tablet camera, for a total number of 273 *query* images. As the shelf images available in *Grocery Products* concern only items belonging to the *Food* macro class, which accounts for 3288 products, we consider also this smaller subset of products, which will be referred to as *Grocery_Food*, whilst *Grocery_Full* will denote all the products of the *Grocery Product* dataset. We depict in Fig. 3 the taxonomy of macro categories that compose the *Grocery_Food* dataset which we are going to use as training set, each categories features several fine grained classes (one for product) not depicted in figure. As for the samples from domain $\mathcal{B}$ needed to train the *discriminator* and *generator* of our architecture (see Section 3.2), we have used 547 additional images cropped from the shelf images available in *Grocery Products*, picked as to have no overlap with the previously mentioned 938 *query* images used at test time. We wish to point out how our formulation requires images from domain $\mathcal{B}$ only for the discriminator of the GAN system. Therefore, few samples without any kind of annotation are sufficient to learn the appearance of products on the shelf. Moreover we can use images from domain $\mathcal{B}$ that depict any kind of product, even items not in $\mathcal{A}$.

Network architectures. For the implementation of DIHE we have used tensorflow[1] as our deep learning framework. For all our test we used as *generator* U-Net (Ronneberger et al., 2015) and as *discriminator* PatchGAN (Isola et al., 2017), the latter producing a dense grid of predictions for each input image. For the *encoder* we tested different available CNN model with or without pretrained weight on the ImageNet-1000 classification task. For the initialization of the *encoder* network on the fine tuning tests we have used the weights publicly available in the tensorflow/models repository.[2] The three network that compose DIHE can easily fit in a single GPU, so training our system, once implemented, in a deep learning framework is straightforward.

Descriptor computation. We will show how for the grocery product recognition scenario, the best performance can be obtained using as embedding the maximum activation of convolution features (MAC Tolias et al., 2016). We extract these descriptors from different layers, concatenate them and finally perform L2-normalization to get a final representation laying on the unit hyper-sphere. For all our tests, we used as distance function $d(X, Y) = 1 - X \cdot Y$ with $X, Y \in \mathcal{D}$ (i.e., one minus the cosine similarity between the two descriptors).

Training details. As for ϕ in L_{reg}, we tried the pixel-wise $L1$ or $L2$ norms, the Structured Similarity Index (SSIM) (Wang et al., 2004) and the Zero Mean Normalized Cross Correlation (ZNCC) and found out the last to work best in all our tests, in the following $\phi(x, y) = ZNCC(x, y)$. The weights of each network are trained to minimize their specific loss functions as introduced in Section 3. We use Adam (Kingma and Ba, 2015) as optimizer with different learning rates for the different tests. Concerning data preprocessing, we use as input color images with fixed size of 256×256 and intensities rescaled between $[-1, 1]$. To obtain the input dimension the original images are rescaled to the target resolution preserving the aspect ratio and filling the extra pixels with 0s. The only additional data augmentation is a preliminary random crop with size at least 80% of the original image to attain the input of the *generator* network.

Evaluation protocol. To test our embedding network we encode all the *reference* images of the considered dataset to create a reference database, then compute the same encoding for the *query* images. For each query vector we perform similarity search against all the reference vectors and retain the K most similar database entries; if the reference image for the product depicted in the query does belong to this set of nearest neighbors we consider recognition to be successful. As a measure of effectiveness of the embedding, we report the accuracy (number of successful recognitions over number of queries) for different K values.

Based on these premises, we train once and for all our architecture using only the reference images belonging to *Grocery_Food*, (i.e., one reference image for each of 3288 different products organized in a multi level hierarchy of products categories). We then use the trained embedding model to address three different test scenarios:

(a) *Grocery_Food*: we recognize the 938 *query* images from *Grocery Products* based on the 3288 reference images from *Grocery_Food*. Thus, all the *reference* images were deployed at training time.
(b) *Grocery_Full*: we recognize the 938 *query* images from *Grocery Products* based on the 8403 reference images from *Grocery_Full*. Thus, only 40% of the *reference* images were deployed at training time.
(c) *Product8600*: we recognize the 273 *query* images cropped from our videos based on the 8597 reference images from *Product8600*. Thus, none of the *reference* images was deployed at training time.

Among the three, (b) is the most likely to happen in practical settings as product appearance changes frequently overtime and it is infeasible to constantly retrain the embedding network, although perhaps a portion of the reference images dealing with the items actually on sale in the store had been used at training time

At test time the *encoder* network is used as a stand alone global image descriptor. Our implementation is quite efficient and can easily encode, on GPU, more than 200 image per second taking into account also the time needed to load the images from disk and rescale them to the 256×256 input size. Finally, given that usually our *reference* database only provides a single image per product and the descriptor dimension is relatively low (between 256 and 1024 floats across different tests), we perform the K-NN similarity search extensively without any kind of approximation. The biggest descriptor database considered in our tests is the one obtained from *Product8600* using a 1024 float dimensional embedding vector (i.e., the *reference* database is a matrix of float with 8600 row and 1024 column). Even on this kind of database given a query descriptor the whole similarity search can be solved in a tenth of second using brute force search, nevertheless the search could be speeded up using KD-Tree or approximate search technique.

4.2. Ablation study

To understand the impact on performance of the different novel components proposed in our architecture, we carry out a model ablation study using as *encoder* PatchGAN (Isola et al., 2017) with MAC features (Tolias et al., 2016) extracted from the last convolutional layer before the output. We use this randomly initialized small network to better highlight the gains provided by the different kind of proposed losses. We will show how to obtain the best performance we rely on a larger pre-initialized network. We train this architecture on the *reference* images of *Grocery_Food* according to six different training losses and report the accuracy dealing with the three test scenarios presented in Section 4 in Table 1. In particular, with reference to the first column, *triplet* denotes training by triplet loss with fixed margin ($\alpha = 0.3$ obtained by cross validation); *hierarchy* denotes training by our triplet loss with variable margin introduced in Section 3.1 ($\alpha_{min} = 0.1, \alpha_{max} = 0.5$); entries with *+GAN* denote deploying the image translation GAN to generate the anchor image $i_a^{\mathcal{B}}$ (Eq. (4): $\lambda_{reg} = 1, \lambda_{emb} = 0$); finally, entries with *+GAN+adv* concerns introducing also the adversarial term in the loss of the GAN generator (Eq. (4): $\lambda_{reg} = 1, \lambda_{emb} = 0.1$). For all the models that do not use GANs, we rely on standard data augmentation techniques (e.g., crop, gaussian blur, color transformation...) to obtain the anchor image given the positive one. In all the tests the networks are randomly initialized and trained for the same number of steps and identical learning rates.

The results reported in Table 1 show how each individual novel component proposed in our DIHE architecture provides a significant performance improvement with respect to the standard triplet ranking loss. Indeed, by comparing rows (2), (4) and (6) to (1), (3) and (5), respectively, it can be observed that modifying the fixed margin of the

[1] https://www.tensorflow.org/.
[2] https://github.com/tensorflow/models/tree/master/research/slim.

Table 1
Ablation study for **DIHE**. Recognition accuracy for 1-NN and 5-NN similarity search in the three considered scenarios. Best results highlighted in bold.

	Training loss	(a) Grocery_Food		(b) Grocery_Full		(c) Product8600	
		K=1	K=5	K=1	K=5	K=1	K=5
(1)	Triplet	0.301	0.430	0.277	0.390	0.351	0.490
(2)	Hierarchy	0.325	0.491	0.302	0.433	0.355	0.553
(3)	Triplet+GAN	0.454	0.626	0.418	0.586	0.512	0.706
(4)	Hierarchy+GAN	0.479	0.660	0.455	0.621	0.538	0.699
(5)	Triplet+GAN+adv	0.470	0.648	0.431	0.595	0.548	0.717
(6)	Hierarchy+GAN+adv (**DIHE**)	**0.481**	**0.688**	**0.463**	**0.642**	**0.553**	**0.732**

standard triplet loss into our proposed hierarchically adaptive margin can improve accuracy with all models and in all scenarios, with a much larger gain in (c) (i.e., completely unseen *reference* images). This proves that embedding a hierarchy into the descriptor space is an effective strategy to help learning an embedding amenable to generalize to unseen data. The main improvements are clearly achieved by methods featuring a GAN network to pursue domain adaptation by generating training samples similar to the images coming from the test domain. Indeed, comparison of (3) and (4) to (1) and (2), respectively, highlights how performance nearly double across all models and scenarios, testifying that the domain shift between test and train data and the lack of multiple training samples are indeed the key issues in this task that the proposed image-to-image translation GAN can help to address very effectively. Finally, comparing (5) and (6) to (3) and (4), respectively, vouches that training the *generator* to produce anchor images not only realistic but also hard to embed turns out always beneficial to performance. Indeed, the adversarial game played by the *generator* and *encoder* may be thought of as an on-line and adaptive hard-mining capable of dynamically synthesize hard to embed samples that help training a more robust embedding. Performance of our overall DIHE architecture are reported in row (6) and show a dramatic improvement with respect to the standard triplet loss, row (1).

4.3. Product recognition

Model and descriptor selection. To ameliorate product recognition performance we can rely on larger networks pre-trained on the ImageNet classification benchmark. To chose the best CNN model as our *encoder* network we downloaded the public available weights of different models trained on ImageNet-1000 classification and test them as general purpose off-the-shelf feature extractors on our datasets without any kind of fine tuning. We considered three different popular CNN models (VGG_16 Simonyan and Zisserman (2014), resnet_[50/101/152] He et al. (2016) and inception_v4 Szegedy et al. (2017)) and compute three kind of different descriptors from activations extracted at various layers:

- **Direct**: directly use the vectorized activation of a given layer. The dimension of the descriptor is the number of elements in the feature maps for that layer.
- **AVG**: perform average pooling on the feature map with a kernel with width and height equals those of the map. Therefore, we use as descriptor the average activation of each convolutional filter for a given layer. The dimension of the descriptor is the number of convolutional filters in the selected layer.
- **MAC** (Tolias et al., 2016): perform max pooling on the feature map with a kernel with width and height equals those of the map. Therefore, we use as descriptor the maximum activation of each convolutional filter for a given layer. The dimension of the descriptor is the number of convolutional filters in the selected layer.

We applied the three different descriptors above at different layers of the three networks and report in Table 2 the 1-NN accuracy using the test protocol described in Section 4.2. For all our tests the descriptors were L2 normalized to unit norm and the similarity search is performed using cosine similarity. In Table 2 we report only some of the best performing layers for each network and additional tests where the descriptors are obtained by concatenation of representations extracted at different depths in the CNN (e.g., *conv4_3+conv5_3* is the concatenation of representations extracted at layers conv4_3 and conv5_3) with L2 normalization performed after concatenation.

Looking at the results in Table 2 we can observe how, in our settings, newer and more powerful CNN, like inception_v4 or resnet_152, fail to achieve the same instance-level distinctiveness of VGG_16. We conjecture that deeper architectures, trained on Imagenet, tend to create more abstract representations that may not provide out-of-the-box features distinctive enough to tell apart many items looking almost identical as required by our problem. Some evidence to support this conjecture may be found in Table 2 due to deeper layers providing typically inferior performance when compared to shallower ones (e.g., VGG_16: conv5_3 vs conv4_3, resnet_50,resnet_101,resnet_152: Block4 vs Block3) Concerning the type of descriptor to use, from Table 2 it seems quite clear that MAC descriptor is the best choice for grocery recognition with respect to AVG or direct activations of fully connected layers.

Given these results, we selected for our fine tuning tests the VGG_16 network with MAC descriptor computed on the concatenation of conv4_3 and conv5_3 layers, and train the overall architecture according to our losses. As the *encoder* is already pre-trained, we perform 5000 iterations of pre-training for the *generator* and *discriminator* (Eq. (4): $\lambda_{reg} = 1, \lambda_{emb} = 0$) before training jointly the whole DIHE architecture. The chosen hyper-parameters obtained by cross validation for the training process are as follows. Learning rates 10^{-5}, 10^{-5} and 10^{-6} for *generator*, *discriminator* and *encoder*, respectively; $\lambda_{emb} = 0.1$, $\lambda_{reg} = 1$, $\alpha_{min} = 0.05$ and $\alpha_{max} = 0.5$.

Before comparing our proposal to other embedding losses, it is interesting to verify whether the improvements provided by the different components (Section 4.2) are valid even when relying on the VGG-16 network pretrained on Imagenet. Purposely, we carry out the same ablation study as in Table 1 and report the results in Table 3. Indeed, the ranking of performances among the different training modalities turns out coherent, although, as expected, the margins are smaller due to the higher performance provided by the baseline.

Comparison with other embedding losses. We compare our architecture to the already mentioned concatenation of MAC descriptors without fine tuning (*MAC*) and to our implementation of different embedding learning methods: Wang et al. (2014) fine tuning using the classic triplet ranking loss (*Triplet*); Hadsell et al. (2006) fine tuning using Siamese networks and the contrastive loss (*Siamese*); Matching networks Vinyals et al. (2016) without the *full context embedding* which does not scale to thousands of classes (*MatchNet*); Zhang et al. (2016) tuplet loss to embed label structure (*Structured*); and Zhang et al. (2017) triplet loss regularized by a spread out term (*Spread*). Similarly to DIHE, all methods are trained on the *reference* images of *Grocery_Food* starting from the very same VGG16 pre-trained on ImageNet-1000 and using the same concatenation of MAC descriptors as embedding function.

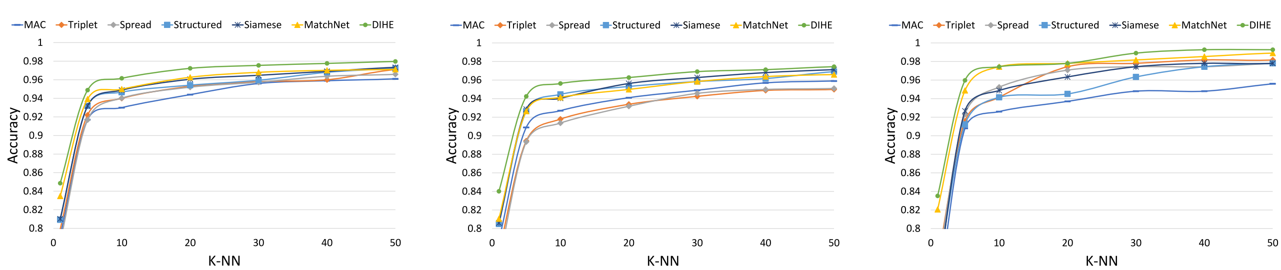
We use the same test scenarios presented in Section 4.2 and report the result in Table 4. As already shown in our model study, MAC

Table 2
1-NN accuracy for different general purpose descriptors obtained from layers of network pre-trained on the ImageNet-1000 classification dataset without any kind of additional fine-tuning. Best results are higlighted in bold.

Network	Layer	Descriptor	Grocery_Food	Grocery_Full	Product8600
VGG_16	conv4_3	MAC	0.789	0.785	0.717
		AVG	0.515	0.510	0.538
	conv5_3	MAC	0.724	0.720	0.611
		AVG	0.406	0.398	0.395
	conv4_3+conv5_3	MAC	**0.792**	**0.787**	**0.725**
		AVG	0.501	0.493	0.523
	fc6	Direct	0.560	0.549	0.549
	fc7	Direct	0.444	0.433	0.432
Inception_v4	Mixed_7a	MAC	0.610	0.603	0.509
		AVG	0.673	0.670	0.560
	Mixed_7b	MAC	0.652	0.641	0.512
		AVG	0.675	0.668	0.5091
	Mixed_7a+Mixed_7b	MAC	0.655	0.646	0.534
		AVG	0.690	0.685	0.542
Resnet_50	Block3	MAC	0.731	0.729	0.703
		AVG	0.441	0.433	0.432
	Block4	MAC	0.654	0.646	0.509
		AVG	0.571	0.558	0.465
	Block3+Block4	MAC	0.723	0.720	0.644
		AVG	0.547	0.538	0.545
Resnet_101	Block3	MAC	0.737	0.735	0.695
		AVG	0.389	0.388	0.432
	Block4	MAC	0.636	0.662	0.490
		AVG	0.570	0.556	0.417
	Block3+Block4	MAC	0.714	0.708	0.626
		AVG	0.535	0.524	0.520
Resnet_152	Block3	MAC	0.708	0.703	0.655
		AVG	0.345	0.337	0.446
	Block4	MAC	0.571	0.561	0.435
		AVG	0.571	0.561	0.435
	Block3+Block4	MAC	0.678	0.671	0.542
		AVG	0.506	0.500	0.504

Table 3
Ablation study for DIHE on a VGG-16 network pretrained on ImageNet-1000. Recognition accuracy for 1-NN and 5-NN similarity search in the three considered scenarios. Best results highlighted in bold.

		(a) Grocery_Food		(b) Grocery_Full		(c) Product8600	
	Training loss	K=1	K=5	K=1	K=5	K=1	K=5
(1)	Triplet	0.799	0.922	0.775	0.894	0.765	0.915
(2)	Hierarchy	0.812	0.933	0.816	0.926	0.805	0.952
(3)	Triplet+GAN	0.829	0.941	0.821	0.937	0.816	0.945
(4)	Hierarchy+GAN	0.832	0.943	0.826	0.933	0.819	0.952
(5)	Triplet+GAN+adv	0.833	**0.948**	0.821	0.937	0.816	0.945
(6)	Hierarchy+GAN+adv (**DIHE**)	**0.853**	**0.948**	**0.842**	**0.942**	**0.827**	**0.959**



(a) **Grocery_Food** (b) **Grocery_Full** (c) **Product8600**

Fig. 4. Accuracy with increasing K in the three scenarios.

activations have strong absolute performance without any need of fine tuning with accuracy at K=1 ranging from 0.72 in the worst case to 0.79 in the best. Starting from such a strong baseline the standard *Triplet* loss is able to only slightly increase performance in scenario (a) and (c) while being slightly penalized in (b), which testifies how in the cross domain and low shot regime is quite hard to properly fine tune an embedding network. The additional regularization term introduced by *Spread* does not seems to help with a slightly decrease in performance across all scenario compared with the standard *Triplet*. *Structured* is the first method to consistently improve performance across all scenario, vouching the importance of deploying label structure in the embedding space for this type of recognition task. *Siamese* based on a simple contrastive loss is able to obtain performance comparable to *Structured* on scenario (a) and (b) while performing slightly worse on the generalization to dataset (c). *MatchNet* is definitely the best competing method for embedding learning in a few shot regime as testified by the

Fig. 5. Qualitative results for K-NN similarity search by DIHE on *Grocery_Food* and *Product8600*. Correct results highlighted in green. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

good improvement obtained with respect to the initial MAC descriptors. Nevertheless, DIHE thanks to the combined use of GAN and hierarchical information is still able to improve the performance, obtaining recognition accuracies for K=1 consistently over 82% across all test. It is worth pointing out that the largest improvement, with respect to the initial MAC results, is obtained on the completely unseen products of the *Product8600*, which vouches for mixing GAN-based domain adaptation and hierarchical embedding to learn an embedding that can generalize very well to unseen items.

In Fig. 4 we report the accuracy of the methods while increasing K. In Figs. 4(a–b) almost all methods converge to more than 95% accuracy for $K > 30$. However, DIHE can provide substantially better results for lower values of K. In Fig. 4(c) DIHE still outperforms the competitors with performance almost equal to those achieved by Matching Networks but showing higher margin against the competitor trained with variant of *triplet ranking loss*.

4.4. Beyond product recognition

To investigate on the generality of our proposal as an improvement over established embedding learning methods, we perform additional tests on the *Office31* benchmark dataset for domain adaptation (Saenko et al., 2010). This dataset consists of 4652 images dealing with 31 classes of office objects acquired in three different domains, namely *Amazon*, concerning ideal images downloaded from the web, *Webcam*, consisting of real images acquired by cheap cameras, and *DSLR*, featuring real images acquired by an high quality camera. Akin to the setup of Grocery Recognition experiments, we use *Amazon* as the *reference* set, thus deploying these images to train the *encoder*, while images from *Webcam* and *DSLR* are used both as training samples for the GAN *discriminator* in DIHE and as two separate *query* sets for the tests. This scenario is compliant to the *full protocol setting* described by Gong et al. (2013): train on the entire labeled source and unlabeled target data and test on annotated target samples. Unfortunately, as the *Office31* dataset

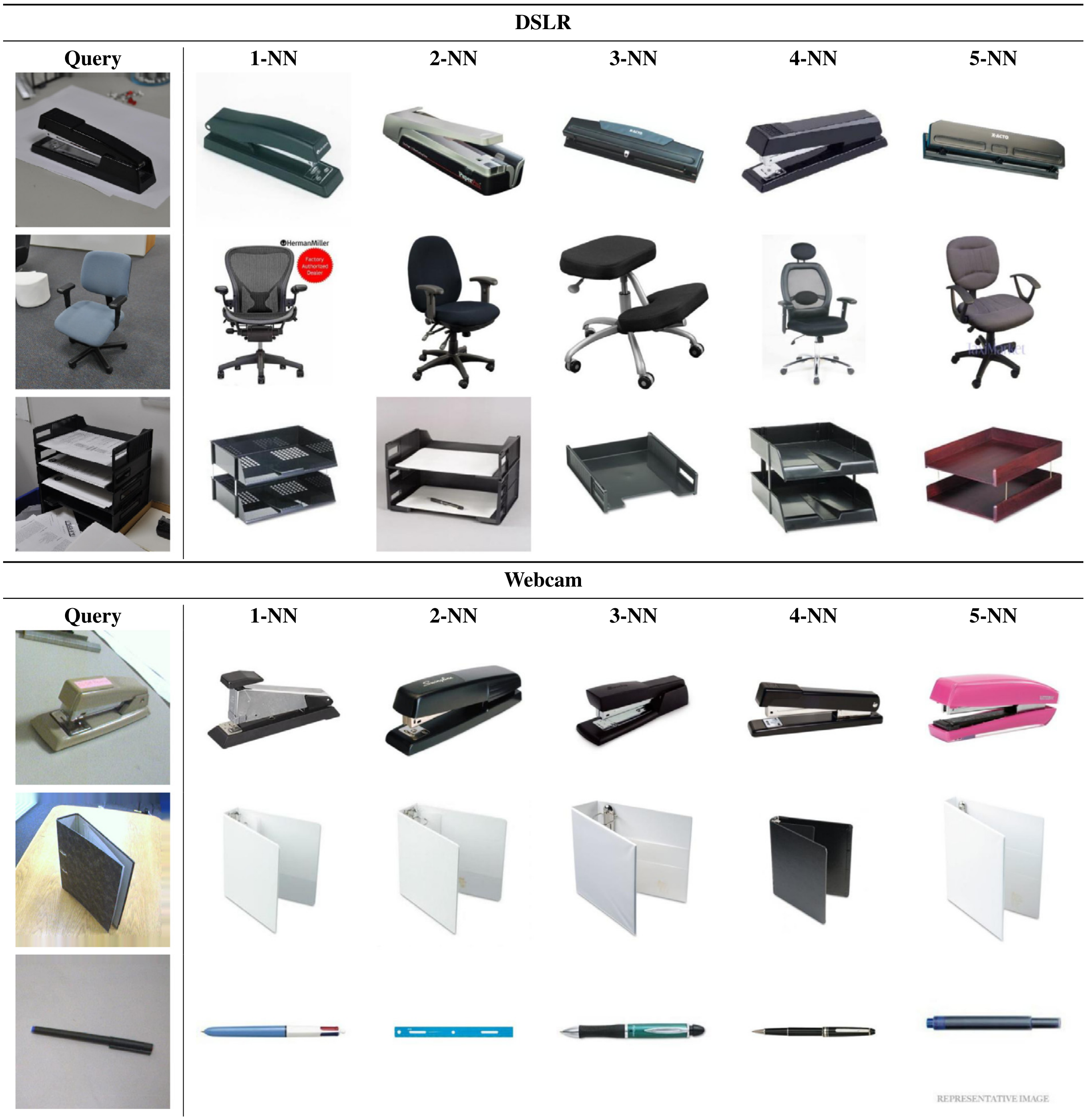


Fig. 6. Qualitative results for K-NN similarity search by DIHE on the *Office31* dataset. Upper portion: *Amazon→DSLR* scenario, lower portion: *Amazon→Webcam*.

does not provide a taxonomy of classes, in these additional experiments DIHE cannot leverage on the hierarchical loss to improve the learned representation. However, unlike the Grocery Recognition scenario, all methods can be trained here by more than one sample per class.

Based on the above described experimental setup, we train different *encoders* starting, once again, from a VGG16 network pre-trained on the ImageNet-1000 dataset. We report the accuracy for 1-NN and 5-NN similarity search in Table 5. To assess the performance of DIHE we compare it once again against the methods considered in Section 4.3, with the exception of *Structured* due to *Office31* lacking a class taxonomy. The first two rows of Table 5 show that, differently from the Grocery Recognition setup, between activations extracted from the pretrained VGG16 network, FC6 outperforms MAC descriptors (computed as in Section 4.3). Coherently with the findings of Table 2, we believe this difference to be due to the office task concerning the recognition of category of objects rather than instances. Accordingly, to obtain the *encoder* network for these tests we substitute the original FC6 layer with a smaller fully connected layer consisting of 512 neurons with randomly initialized weights and then perform fine tuning on the *Amazon* images.

In this different experimental settings wherein more training samples per class are available, both *Siamese*, *Spread Out* and *Matchnet* are outperformed by the plain *Triplet* ranking loss of Wang et al. (2014), which turns out the best performing established method achieving a 1-NN accuracy of 59% and 62% for test datasets *Webcam* and *DSLR*, respectively. Yet, thanks to the introduction of the *Generator* in the training loop, DIHE can yield a significant performance improvement reaching a 1-NN accuracy of 62% (+3%) and 66% (+4%) for *Webcam* and *DSLR*, respectively, with best or comparable 5-NN accuracy when compared with competing methods using descriptors with the same dimension. Moreover, our proposal can outperform the descriptor extracted by FC6 that is twice larger on 3 experiments out of 4, obtaining comparable performance on the fourth.

Although performance on *Office31* turns out well below the state-of-the-art attainable by image recognition methods based on classifiers, amenable to handle a fixed and possible small number of

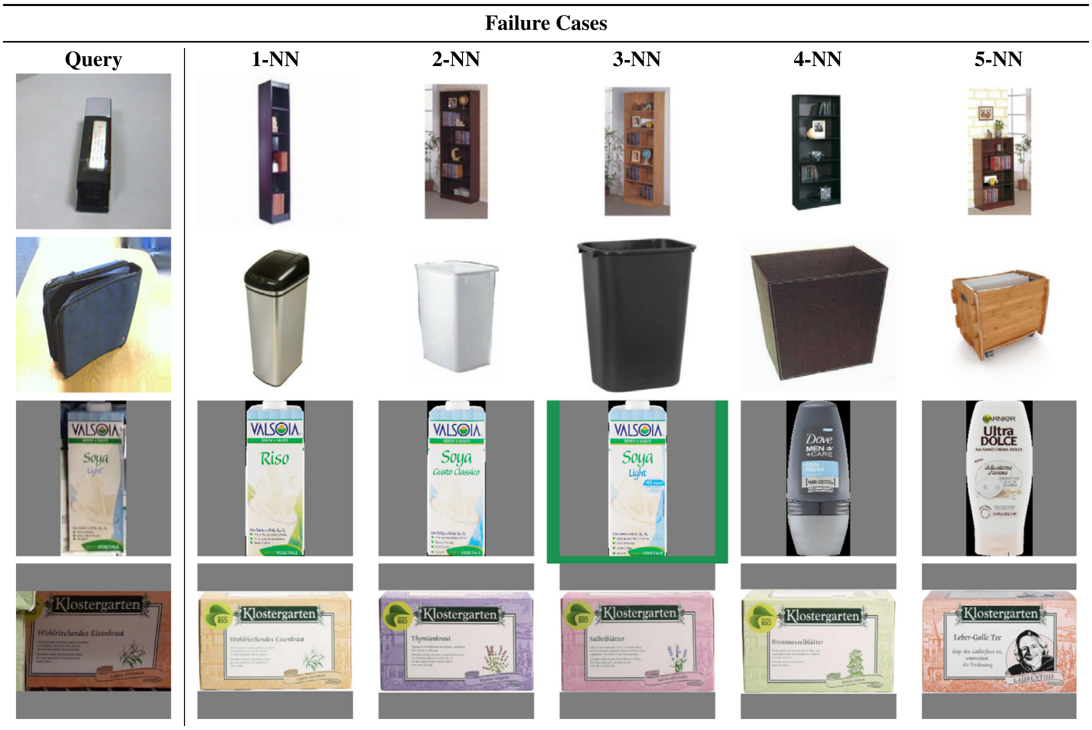


Fig. 7. Some wrong recognitions yielded by DIHE on the different datasets.

Fig. 8. Images generated by the GAN trained jointly with the embedding network in DIHE for the *Office31-Amazon→Webcam* (left) and *Grocery_Food* (right) scenarios. Columns labeled as *Original* depict images provided as input to the *Generator* while those labeled as *Generated* show the corresponding outputs.

classes, we argue that the experiments reported in this Section further highlight the advantages provided by our proposed DIHE architecture with respect to common feature learning approaches.

4.5. Qualitative results

Figs. 5 and 6 report some successful recognitions obtained by K-NN similarity search based on DIHE. The upper portion of Fig. 5, dealing

Table 4
Recognition accuracy for 1-NN and 5-NN similarity search in the three considered scenarios. Best results highlighted in bold, differences between DIHE and best performing competitor reported in the last line.

	(a) Grocery_Food		(b) Grocery_Full		(c) Product8600	
Descriptor	K=1	K=5	K=1	K=5	K=1	K=5
MAC	0.792	0.917	0.787	0.9093	0.725	0.908
Triplet (Wang et al., 2014)	0.799	0.922	0.775	0.894	0.765	0.915
Spread (Zhang et al., 2017)	0.784	0.916	0.764	0.893	0.758	0.923
Structured (Zhang et al., 2016)	0.809	0.931	0.804	0.926	0.750	0.912
Siamese (Hadsell et al., 2006)	0.810	0.931	0.805	0.928	0.733	0.926
MatchNet (Vinyals et al., 2016)	0.834	0.939	0.810	0.929	0.820	0.948
DIHE	**0.853**	**0.948**	**0.842**	**0.942**	**0.827**	**0.959**
	+0.02	+0.01	+0.03	+0.02	+0.007	+0.01

Table 5
Recognition accuracy for 1-NN and 5-NN similarity search on two subset of the *Office31* using as *reference* images the *Amazon* subset. Best resulted highlighted in bold, differences between DIHE and best performing competitor reported in the last line.

	(a) Webcam		(b) DSLR	
Descriptor	K=1	K=5	K=1	K=5
FC6	0.470	**0.698**	0.489	0.712
MAC	0.382	0.640	0.393	0.660
Triplet (Wang et al., 2014)	0.596	0.675	0.628	0.710
Spread (Zhang et al., 2017)	0.591	0.660	0.590	0.670
Siamese (Hadsell et al., 2006)	0.469	0.547	0.576	0.630
MatchNet (Vinyals et al., 2016)	0.522	0.579	0.566	0.618
DIHE	**0.628**	0.691	**0.662**	**0.742**
	+0.03	−0.007	+0.03	+0.03

with the *Grocery_Food* scenario, shows query images for products quite hard to recognize due to both the *reference* database featuring several items looking remarkably similar as well as nuisances like shadows and partial occlusions (first query) or slightly deformed products (third query). The lower portion of Fig. 5 concerns the *Product8600* scenario: as clearly highlighted by the third and second query, despite differences between items belonging to the same brand and category (e.g., the pasta boxes in the last row) being often subtle, DIHE can recognize products correctly. In Fig. 6 we report some results dealing with the *Office-31* dataset which vouch how DIHE can correctly retrieve images depicting objects belonging to the same category as the query. In Fig. 7, we highlight some failure cases. In the first row DIHE wrongly recognizes a stapler as a bookshelf, whilst in the second a ring binder is mistaken as a trash bin. In the third row DIHE seems to recognize the macro class and brand of the query product though missing the correct instance level label (i.e., the correct NN, highlighted in green, is retrieved as the 3-NN). A similar issue pertains the fourth query image: all the first 5-NNs belong to the *Tea* macro class, but the correct item is not ranked among them.

Finally, in Fig. 8 we show some training images generated by our GAN framework to pursue domain invariance (Section 3.2). It is worth observing how the training samples created by our GAN seem to preserve both the overall structure and details of the input images, which is very important when addressing instance level recognition between many similarly looking items, while modifying significantly the brightness and colors and injecting some blur, thereby realizing a domain translation between the input and output images.

5. Conclusion

The experimental results demonstrate how our proposed hierarchical modification to the triplet ranking loss is effective in learning embedding functions that generalize better to unseen data. Moreover, the integration of a GAN at training time, so as to obtain the whole DIHE architecture, turns out a very effective approach in scenarios where only few/one samples per class are available at training time. Indeed, DIHE allows for learning a representation not only robust to domain shift but also better all around due to the *Generator* network effectively producing potentially infinite hard training samples. DIHE was designed to solve the task of Grocery Product recognition, in which it can deliver remarkably good performance. Nonetheless, experiments in different scenarios suggest that our architecture may be effectively deployed in settings that features challenges such as few training samples per class, different domains between train and test data and a taxonomy among classes to be recognized. The results of the ablation study of Section 4.2 show clearly how the main improvement in performance is provided by the original introduction of a GAN network trained end-to-end together with the embedding network in order to augment the training set. In the future we would like to further investigate on this novel concept through different combination of embedding losses and GAN architectures. For example, a possible variant of DIHE may concern a GAN that would operate at the feature embedding level rather than at image level, thereby hallucinating embedding vectors from the domain $\mathcal{B}$ given those from domain $\mathcal{A}$, with these vectors amenable to compute any kind of embedding loss.

Finally, in this paper we have proposed an architecture to recognize a product item extracted from a shelf image, though we did not address how to actually detect the individual product items within such an image. Recent works like Qiao et al. (2017) have shown how a region proposal CNN can be successfully trained to extract bounding boxes surrounding grocery products from an image featuring the whole shelf. Thus, Qiao et al. (2017) and DIHE may be combined effortlessly in a complete pipeline that given a single shelf image and one reference image for each product can detect and recognize the displayed items.

Acknowledgment

We would like to thank *Centro Studi s.r.l.* for funding this research project.

References

Advani, S., Smith, B., Tanabe, Y., Irick, K., Cotter, M., Sampson, J., Narayanan, V., 2015. Visual co-occurrence network: Using context for large- scale object recognition in retail. In: Embedded Systems For Real-time Mul- timedia (ESTIMedia). In: 2015 13th IEEE Symposium on, IEEE, pp. 1–10.

Baz, I., Yoruk, E., Cetin, M., 2016. Context-aware hybrid classification system for fine-grained retail product recognition. In: Image, Video, and Multidimensional Signal Processing Workshop, IVMSP. In: 2016 IEEE 12th, IEEE, pp. 1–5.

Cotter, M., Advani, S., Sampson, J., Irick, K., Narayanan, V., 2014. A hardware accelerated multilevel visual classifier for embedded visual-assist systems. In: Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design. IEEE Press, pp. 96–100.

Franco, A., Maltoni, D., Papi, S., 2017. Grocery product detection and recognition. Expert Syst. Appl..

George, M., Floerkemeier, C., 2014. Recognizing products: A per-exemplar multi-label image classification approach. In: Computer Vision–ECCV 2014. Springer, pp. 440–455.

Gong, B., Grauman, K., Sha, F., 2013. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In: International Conference on Machine Learning. pp. 222–230.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. Adv. Neural Inf. Process. Syst. 2672–2680.

Gordo, A., Almazán, J., Revaud, J., Larlus, D., 2016. Deep image retrieval: Learning global representations for image search. In: European Conference on Computer Vision. Springer, pp. 241–257.

Hadi Kiapour, M., Han, X., Lazebnik, S., Berg, A.C., Berg, T.L., 2015. Where to buy it: Matching street clothing photos in online shops. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3343–3351.

Hadsell, R., Chopra, S., LeCun, Y., 2006. Dimensionality reduction by learning an invariant mapping. In: Computer Vision and Pattern Recognition. In: 2006 IEEE Computer Society Conference on, IEEE, pp. 1735–1742.

Hariharan, B., Girshick, R., 2017. Low-shot visual recognition by shrinking and hallucinating features, In: Proc. of IEEE Int. Conf. on Computer Vision, ICCV, Venice, Italy.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.
Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In: Computer Vision and Pattern Recognition, CVPR 2017.
Karlinsky, L., Shtok, J., Tzur, Y., Tzadok, A., 2017. Fine-grained recognition of thousands of object categories with single-example training. In: Proceedings Of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4113–4122.
Kingma, D., Ba, J., 2015. Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference for Learning Representations.
Merler, M., Galleguillos, C., Belongie, S., 2007. Recognizing groceries in situ using in vitro training data. In: Computer Vision and Pattern Recognition, 2007. CVPR'07. In: IEEE Conference on, IEEE, pp. 1–8.
Paolanti, M., Sturari, M., Mancini, A., Zingaretti, P., Frontoni, E., 2017. Mobile robot for retail surveying and inventory using visual and textual analysis of monocular pictures based on deep learning. In: Mobile Robots (ECMR). In: 2017 European Conference on, IEEE, pp. 1–6.
Qiao, S., Shen, W., Qiu, W., Liu, C., Yuille, A., 2017. Scalenet: Guiding object proposal generation in supermarkets and beyond. In: The IEEE International Conference on Computer Vision, ICCV.
Qiu, Z., Pan, Y., Yao, T., Mei, T., 2017. Deep semantic hashing with generative adversarial networks. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 225–234.
Radford, A., Metz, L., Chintala, S., 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In: International Conference on Learning Representation, ICLR 2016.
Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 234–241.
Saenko, K., Kulis, B., Fritz, M., Darrell, T., 2010. Adapting visual category models to new domains. In: Computer Vision–ECCV 2010. pp. 213–226.
Santarcangelo, V., Farinella, G.M., Furnari, A., Battiato, S., 2018. Market basket analysis from egocentric videos. Pattern Recognit. Lett..
Schroff, F., Kalenichenko, D., Philbin, J., 2015. Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 815–823.
Shankar, D., Narumanchi, S., Ananya, H., Kompalli, P., Chaudhury, K., 2017. Deep learning based large scale visual recommendation and search for e-commerce. arXiv preprint arXiv:1703.02344.
Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S., 2014. CNN features off-the-shelf: An astounding baseline for recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 806–813.
Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R., 2017. Learning from simulated and unsupervised images through adversarial training. In: Computer Vision and Pattern Recognition, CVPR 2017.
Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556.
Snell, J., Swersky, K., Zemel, R., 2017. Prototypical networks for few-shot learning. Adv. Neural Inf. Process. Syst. 4077–4087.
Sturari, M., Liciotti, D., Pierdicca, R., Frontoni, E., Mancini, A., Contigiani, M., Zingaretti, P., 2016. Robust and affordable retail customer profiling by vision and radio beacon sensor fusion. Pattern Recognit. Lett. 81, 30–40.
Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A., 2017. Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning. AAAI, pp. 4278–4284.
Tolias, G., Sicre, R., Jégou, H., 2016. Particular object retrieval with integral max-pooling of CNN activations. In: Procedings Of the International Conference on Learning Representations.
Tonioni, A., Di Stefano, L., 2017. Product recognition in store shelves as a sub-graph isomorphism problem. In: International Conference on Image Analysis and Processing. Springer, pp. 682–693.
Tzeng, E., Hoffman, J., Saenko, K., Darrell, T., 2017. Adversarial discriminative domain adaptation. In: Computer Vision and Pattern Recognition, CVPR. p. 4.
Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al., 2016. Matching networks for one shot learning. Adv. Neural Inf. Process. Syst. 3630–3638.
Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process 13, 600–612.
Wang, S., Fidler, S., Urtasun, R., 2015. Lost shopping! monocular localization in large indoor spaces. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2695–2703.
Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y., 2014. Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1386–1393.
Wang, X., Sun, Z., Zhang, W., Zhou, Y., Jiang, Y.G., 2016. Matching user photos to online products with robust deep features. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval. ACM, pp. 7–14.
Wu, C.Y., Manmatha, R., Smola, A.J., Krähenbühl, P., 2017. Sampling matters in deep embedding learning. In: The IEEE International Conference on Computer Vision, ICCV.
Zhang, X., Yu, F.X., Kumar, S., Chang, S.F., 2017. Learning spread-out local feature descriptors. In: The IEEE International Conference on Computer Vision, ICCV.
Zhang, X., Zhou, F., Lin, Y., Zhang, S., 2016. Embedding label structures for fine-grained feature representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1114–1123.
Zhu, J.Y., Park, T., Isola, P., Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: International Computer Vision Conference, ICCV 2017.