Powering Robust Fashion Retrieval with Information Rich Feature Embeddings

Ayush Chopra*, Abhishek Sinha*, Hiresh Gupta*, Mausoom Sarkar*, Kumar Ayush, Balaji K
Media and Data Science Research
Adobe Inc.
(ayuchopr, abhsinha, hiregupt, msarkar, kayush, kbalaji)@adobe.com

Abstract

Visual content-based product retrieval has become increasingly important for e-commerce. Fashion retrieval, in particular, is a challenging problem owing to a wide range of visual distortions in their product images. In this paper, we propose a Grid Search Network (GSN) for learning feature embeddings for fashion retrieval. The proposed approach posits the training procedure as a search problem, focused on locating matches for a reference query image in a grid containing both positive and negative images w.r.t the query. The proposed framework significantly outperforms existing state-of-the-art methods on benchmark fashion datasets. We also utilize a reinforcement learning based strategy to learn a specialized transformation function which further improves retrieval performance when applied over the feature embeddings. We also extend the reinforcement learning based strategy to learn custom kernel functions for SVM based classification over FashionMNIST and MNIST datasets, showing improved performance. We highlight the generalization capabilities of this search strategy by showing performance improvement in domains beyond fashion.

1. Introduction

The 2018 Fashion United [21] analysis values the global fashion industry at $3 trillion, an approximate 2% of the world GDP. The PWC retail newsletter from the same year estimates an annual growth rate (CAGR) of 7.6% for the worldwide fashion commerce market [13] with the projected revenue for apparel and clothing segment itself to rise by $257.8 billion over the next 2 years [16]. The State of Fashion report by McKinsey [11] in 2018 emphasizes the critical role of intelligent systems in driving this growth.

Consequently, several recent efforts have been directed towards robust clothing retrieval [5] [24] [6] [25] [10] [22]. Generating robust visual embeddings for fashion retrieval is a particularly challenging problem owing to a wide variety of visual distortions and deformable nature of clothing items.

Figure 1. Input to the proposed Grid Search Network includes a Query Image and a Target Grid consisting of positive and negative images with respect to the Query Image.

Deep learning models have performed remarkably well in several domains such as computer vision [8, 2], natural language processing [3, 17], and speech recognition [12]. In the present work, we introduce a deep learning based grid search network to learn robust visual embeddings to facilitate fashion retrieval. The proposed framework posits the training procedure as a search problem w.r.t a reference query image, focused on identifying all matches from a target grid containing positive and negative samples. To achieve this, we optimize a grid search loss adapted from a double hinge contrastive objective to enable concurrent comparison of multiple feature embeddings corresponding to image instances in the target grid with the reference query image. A sample query image and target grid pair is shown in Figure 1. The 3×3 target grid here consists of 2 positive samples (in green boxes) and 7 negative samples w.r.t the query image. A positive image for the query image corresponds to the same product as the query and a negative image corresponds to a different product (from the same or different category) than the query.

We also use deep reinforcement learning to learn a com-

*Equal Contribution

posite function for element-wise transformation of the embeddings from our grid search network for improved retrieval performance. Deep reinforcement learning based search mechanisms have been recently used for discovering neural optimization methods [1], neural activation functions [14] and neural architecture designs [26]. In this work, we build on these existing works allowing for the automatic discovery of a function for element-wise transformation of the learned visual embeddings. Numerical experiments show that such an approach is able to discover a specialized transformation function that is more efficient and the transformed feature embeddings achieve better retrieval performance on thhe considered datasets.

We also propose to use the automated search techniques to discover custom kernel functions for SVM based FashionMNIST [23] and MNIST [4] classification. Using the reinforcement learning-based search approach, we find a custom kernel function (by using the data of a particular problem) that shows promising performance.

Our main contributions in this work are summarized below:

- We introduce a novel Grid Search Network (GSN) framework for generating robust visual embeddings.
- We validate the effectiveness of GSN for fashion retrieval by achieving state-of-the-art results on benchmark fashion datasets - Inshop retrieval and Consumer-to-Shop retrieval datasets [10].
- We leverage a reinforcement learning based search strategy to learn composite functions to be used as element-wise transformations and kernel functions in improving performance on retrieval and SVM classification respectively.

2. Related Work

2.1. Fashion Retrieval

Triplet and Contrastive losses have proven to be successful in several recent works [5] [24] [6] [25]. Triplets are defined as a combination of a reference input, a positive example and a negative example. Networks are trained with the objective of minimizing distance of the reference image from the positive sample and maximizing distance of the reference image from the negative sample.

FashionNet[10] augments visual content with landmark and category annotations and trains on a triplet objective. Generating annotations, however, requires a lot of supervision thereby significantly limiting its practical usability.

VAM [22] trains a two-stream network with an attention branch and a global convolutional branch, and then concatenates the generated vectors to optimize a standard triplet objective. However, it requires additional semantic class annotations as prior for pre-training attention maps.

HDC [24] uses an ensemble of models with increasing complexities in a cascaded manner to mine negative samples at different levels during training. Feature vectors coming from each sub-network are scaled by constant weights and concatenated to generate representations which are used for retrieval.

HTL [25] introduces an adversarial approach for mining hard negative using GAN and optimizes a standard triplet objective.

In the present work, we posit the training procedure as a search problem. We propose a grid framework that leverages concurrent comparison of the reference input against multiple negative and positive samples to produce robust feature embeddings for improved retrieval performance. The proposed grid search loss is trained with an interleaved offline sampling procedure for mining of hard negatives. Variable hard negative samples are introduced into the grid in a probabilistic manner during training iterations. Further, instead of trivial concatenation of feature vectors, we learn transformations over these vectors, to generate representations that improve retrieval performance.

2.2. Learning Feature Transformations

We use automated search techniques to discover composite functions for element-wise transformation of the learned feature embeddings to improve fashion retrieval performance. To this end, we use deep reinforcement learning based search mechanisms that have recently been used for discovering neural optimization methods [1], neural activation functions [14] and neural architecture designs [26]. For example, [26] uses a recurrent neural network to generate the model descriptions of neural networks and trains this RNN with reinforcement learning to maximize the expected accuracy of the generated architectures on a validation set. We follow a similar approach and use an RNN to generate functions (to be used for element-wise transformation of feature embeddings) and use reinforcement learning to train the RNN to maximize top-1 retrieval performance. We verify the effectiveness of our approach by conducting an empirical evaluation with the discovered transformation function. Our experiments show that the learned transformation function shows improvement on the retrieval performance.

3. Proposed Framework

3.1. Network Inputs

The proposed Grid Search Network (GSN) architecture consists of 2 inputs, a query image and a target grid. The query image is of shape $227 \times 227 \times 3$. The target grid is of shape $(K*227) \times (K*227) \times 3$. The target grid is generated

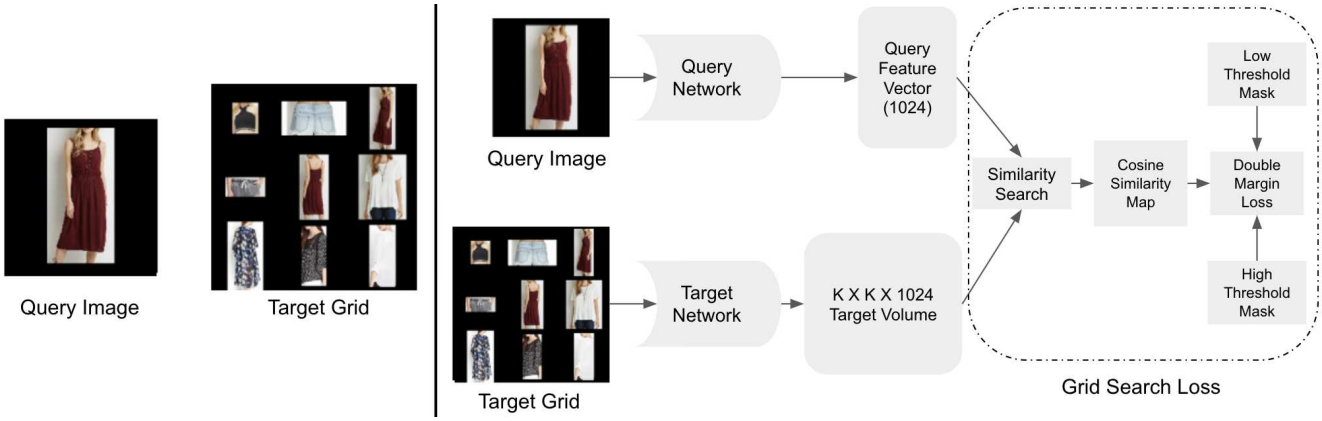


Figure 2. GSN (a) Network Inputs (b) Network Architecture and Search Loss. For the grid here, K=3. Cosine Map, Low Mask and High Mask all have shape $K \times K$. The target network is detailed in Figure 3

by randomly sampling positive and negative samples w.r.t to the query image and arranging them into a $K \times K$ grid of images. The number of positive and negative samples are determined using:

$$\#positives = min(rand(1, K^2 - 1),\ maxPos) \quad (1)$$

$$\#negatives = K^2 - \#positives \quad (2)$$

Here, $maxPos$ is the maximum number of positive matches for the reference query image in the train dataset. Sample inputs are presented in Figure 1 where the grid has 2 positive samples and 7 negative samples w.r.t the reference query image.

3.2. Network Architecture

The network architecture is visualized in Figure 2. The grid search network is characterized by two trunks, a query trunk and a target trunk. Both query and target trunks have Inception-v1 [20] as the base architecture. The target trunk base architecture can be substituted with a fully convolutional trunk that has output_stride=32. The target trunk network is visualized in Figure 3. The query and target trunks do not share weights. This design choice is validated empirically. The input for the query trunk is the query image and for the target trunk is the target grid. The query trunk generates a 1024-dim query vector. For a $(K*227)\times(K*227)\times 3$ target grid, the target trunk generates a $K \times K \times 1024$ target volume. The target trunk is specifically designed to ensure receptive field consistency such that each depth vector of the output target volume corresponds to a single image of the input grid. We validate this design through experiments discussed in Section 5.

3.3. Grid Search Loss

The idea of search is to scan through a set of items and concurrently figure out what is similar and what is not. We incorporate this by having a query image and a grid of $K \times K$ search images. To do this, we L2-normalize the vectors and then perform a similarity search operation (Equation 3). We carry out this search similarity operation by performing a convolution of the $1 \times 1 \times d$ query vector (u), over the $K \times K \times d$ feature volume (v). This search operation generates a $K \times K$ similarity grid, that consists of similarity scores between the query image and each of the corresponding target grid image. Equation 4 describes the final grid search loss computation, on the similarity grid S, and the corresponding low and high threshold masks, L and H, for the query-grid pair.

$$S(u, v) = u \otimes v \quad (3)$$

$$loss = \sum_{i=1}^{k}\sum_{j=1}^{k}(max(0,\ L[i,j] - S[i,j])^2 + max(0, S[i,j] - H[i,j])^2) \quad (4)$$

As in [9] [15], we use low and high thresholds to prevent the distance between embeddings of similar instances from collapsing to zero. Our formulation of these threshold masks, as shown in Figure 2, provides a range for oscillation of cosine similarities during training. For a $K \times K$ target grid, the low and high threshold masks are also of shape $K \times K$ where each element corresponds to the specific threshold value for each of the images in target grid. We empirically determine the low thresholds for similar and dissimilar image instances to be 0.9 and -1 respectively and the high thresholds to be 1 and 0.2 respectively.

The grid search loss is trained with an interleaved offline sampling procedure where hard negatives are mined for query images at regular intervals. These hard negatives are sampled in a probabilistic manner into the target grid for subsequent training iterations.

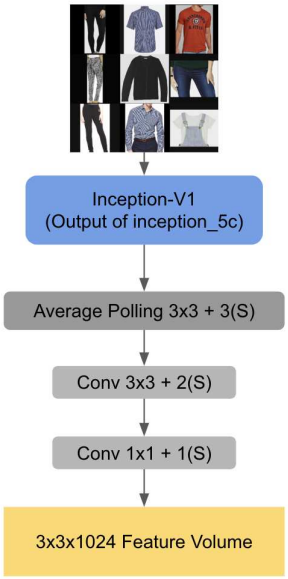


Figure 3. Network architecture of target trunk. For a $K \times K$ target grid, this target network generates a target volume of shape $K \times K \times 1024$. The input target grid in this figure has $K = 3$.

3.4. Learning Feature Transformations

We use automated search techniques to discover composite functions for element-wise transformation of the learned feature embeddings to improve the retrieval performance. The fully convolutional nature of our target network allows us to extract features of a single image ($227 \times 227 \times 3$) from the target trunk. Therefore, we can extract two feature vectors for the same image, one coming from query trunk and another from the target grid trunk. Now, rather than just concatenating these different feature vectors of a single image as in [24], we propose to learn a composite function for element-wise transformation of the feature vectors and use that as the combined feature representation of the image to boost the retrieval performance. We also employ a similar approach to discover functions to be used as kernels for SVM based FashionMNIST and MNIST classification.

Here, we explain our approach for automated discovery of functions. As discussed, we built upon the existing reinforcement learning based search mechanisms [1, 14, 26].

As shown in Figure 4, the function is constructed by repeatedly composing the "core unit". A core unit first selects two operands ($op1$ and $op2$), then two unary functions ($u1$ and $u2$) to apply on the operands and finally a binary function b that combines the outputs of the two unary functions. The resulting $b(u1(op1), u2(op2))$ then becomes an operand that can be selected in the next group of predictions. Every prediction is carried out by a softmax classifier and then fed into the next time step as input.

Given the search space, the goal of the search algorithm is to find effective choices for the unary and binary functions. We use an RNN controller [26]. At each timestep, the controller predicts a single component of the function. The prediction is fed back to the controller in the next timestep, and this process is repeated until every component of the function is predicted. For the purpose our experiments, we consider only two core units. Once a candidate function has been generated by the search algorithm, it is

- used for element-wise transformation of the learned feature embeddings in case of fashion retrieval. The top-1 retrieval performance using the transformed feature embeddings is used as a reward signal to train the RNN controller.
- used as a custom kernel to train an SVM for some task, such as FashionMNIST/MNIST classification. After training, the validation accuracy is recorded and used as a reward signal to train the RNN controller. This training pushes the controller to generate kernel functions that have high validation accuracies.

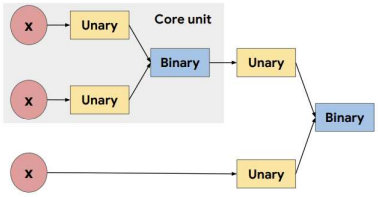
The controller is trained via reinforcement learning using the policy gradient obtained via REINFORCE[19] algorithm. The operands and functions accessible for learning the composite function for retrieval task (transformation function) and SVM classification (custom kernel function) are discussed in Section 6.2.1 and 6.2.2.

We also highlight the generic applicability of the method to improve recall rate performance on Inshop Retrieval, Consumer-to-Shop Retrieval [10] and CARS [7] benchmark datasets using GSN and [24] approaches.

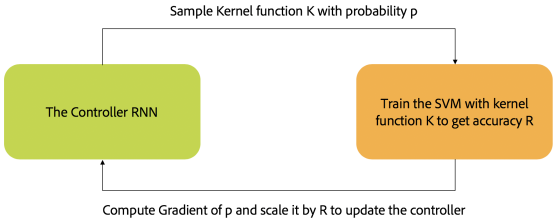
4. Dataset

We conduct fashion retrieval experiments on the Consumer-to-Shop and Inshop retrieval datasets from the DeepFashion benchmark [10]. **Consumer-to-Shop retrieval** contains $2,39,557$ consumer/shop cloth images across $33,881$ clothing items. The presence of large amounts of occlusion, deformation, illumination variations and blur in the images makes the problem particularly challenging. **Inshop retrieval** contains $52,712$ images across $7,982$ clothing items clicked within a curated shop environment. The images are characterized by large variations in pose and scale. To ensure consistency in comparison, we follow the same train-val-test splits as used in previous works for all our experiments.

To test the generalizability of our reinforcement learning based strategy, we learn element-wise transformations over the features obtained by [24] for **Standard CARS** dataset [7] and show improved retrieval performance. We also show the applicability of the proposed RL approach by learning custom kernel functions for SVM based classification over FashionMNIST and MNIST datasets and show improved performance.



(a) Example Function Structure



(b) Overview of Function Search

Figure 4. The left figure shows an example composite function structure. The function is composed of multiple repetitions of the "core unit", which consists of two inputs, two unary functions, and one binary function. Unary functions take in a single scalar input and return a single scalar output. Binary functions take in two scalar inputs and return a single scalar output. The figure in right shows an overview of the search mechanism.

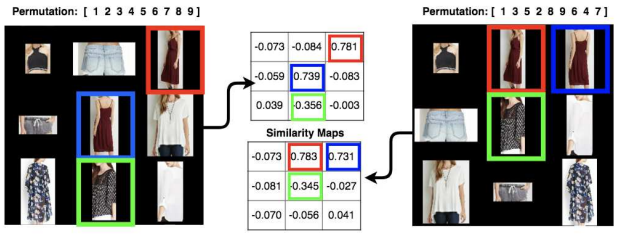


Figure 5. Cosine similarity for two configurations of the target grid with the query specified in Figure 2(a).

5. Qualitative Analysis

The convolution layers in the target trunk are designed to ensure that each vector along the depth in the target volume focuses on a single image from the target grid. We posit that this mapping of the target grid images and depth vectors in the output volume is critical to the effectiveness of our grid search loss. We validate the target trunk architecture empirically. For the query image and the target grid in Figure 2, we permute the arrangement of images in the grid and generate multiple configurations. Using our network, we compute similarity map of the query image with each of the target grid configuration. In Figure 5, we display two sample permutations and the corresponding similarity maps generated with the query image (in Figure 2a). The consistency of values for grid images across different grid locations highlights the sanity of our target trunk architecture.

6. Experiments

Experiments were conducted on 4 NVIDIA GTX 1080Ti GPU's with 12 GB memory on a machine equipped with 64 GB CPU memory. Experiments were set up in tensorflow 1.10. Hyperparameters used: batch size = 8, iteration size = 8, learning rate = 0.0045, optimizer = SGD with momentum (= 0.9), K = 3.

6.1. Fashion Retrieval

We conduct experiments on the Consumer-to-Shop and Inshop retrieval datasets from the DeepFashion benchmark [10]. Network architecture and search loss have been discussed in detail in Section 3. For our experiments, we set K = 3. We use top-n retrieval to measure performance, such that a successful retrieval is counted if at least one exact fashion item has been found in the top n retrieved results. Comparison against existing state-of-art methods is summarized in Section 7.1. At inference time, we use input image of resolution $227 \times 227 \times 3$ and cosine similarity as the distance metric.

6.2. Learning Feature Transformations

To validate the effectiveness of our proposed strategy to learn functions, we conduct experiments on tasks of visual retrieval (element-wise feature transformation) and classification (custom kernel functions for SVM based classification). We setup the technical background previously in Section 3.4 . We describe the operands and functions accessible for learning the composite function for the retrieval and SVM classification tasks in the following two sub-sections.

6.2.1 Retrieval

The operands, unary, and binary functions accessible to the RNN controller are the following -

- Operands using the vectors from query trunk and target trunk, x and y: x, y, $x + y$
- Unary functions: x, $-x$, x^2, $|x|$, x^3, $\sqrt{(|x|)}$, e^x, $\sin x$, $\cos x$, $\sinh x$, $\cosh x$, $\tanh x$, $\sqrt{x}$,$||x||_1$, $||x||_2$, $erfx$, $\tan^{-1} x$, $\sigma(x)$, $\max(x, 0)$, $\min(x, 0)$, $\log_e(1 + e^x)$
- Binary functions: $x_1 + x_2$, $x_1 - x_2$, $x_1 \cdot x_2$, $x_1 * x_2$, $\max(x_1, x_2)$, $concat(x_1, x_2)$, $\min(x_1, x_2)$, $\sigma(x_1) * x_2$, x_1

Dataset	In-shop					Cons-2-Shop				
Config/Top-N	@1	@5	@10	@20	@50	@1	@5	@10	@20	@50
GSN[Q]	0.846	0.917	0.935	0.951	0.966	0.19	0.28	0.33	0.40	0.49
GSN[T]	0.853	0.925	0.943	0.960	0.972	0.21	0.30	0.35	0.42	0.51
GSN[$Q \oplus T$]	0.861	0.937	0.952	0.968	0.980	0.23	0.33	0.38	0.45	0.55
GSN[RL]	0.870	0.946	0.959	0.975	0.986	0.25	0.35	0.41	0.47	0.57

Table 1. Ablation Study: top-n recall comparison of different embedding representations from the Grid Search Network.

We highlight the generic applicability of the method to improve recall rate performance on Inshop Retrieval, Consumer-to-Shop Retrieval [10] and CARS196 [7] benchmark datasets using GSN and [24] approaches.

To conduct experiments with [24], we directly sample feature vectors from their open-source models [1] [2] without any additional training of their embedding network. Learning feature transformations using our RL strategy on these representations leads to performance improvement as discussed in results.

6.2.2 Classification

The operands, unary functions and binary functions that are accessible to the RNN controller in case of SVM kernel discovery are the following:

- Operands using the two vectors x and y: $|x-y|$, $(x+y)$, $(x-y)^2$, $||x-y||_1$, $||x-y||_2$, $x \cdot y$, $x * y$, γ[3]
- Unary functions: x, $-x$, x^2, $|x|$, x^3, $\sqrt{(|x|)}$, e^x, $\sin x$, $\cos x$, $\sinh x$, $\cosh x$, $\tanh x$, $||x||_1$, $||x||_2$, $\sigma(x)$, $\max(x, 0)$, $\min(x, 0,)\log_e(1+e^x)$
- Binary functions: $x_1 + x_2$, $x_1 - x_2$, $x_1 \cdot x_2$, $x_1 * x_2$, $\max(x_1, x_2)$, $\min(x_1, x_2)$, $e^{-|x_1-x_2|}$, x_1

The operands have been chosen in a way so as to keep the discovered kernel function symmetric. We conduct experiments over FashionMNIST and MNIST datasets.

7. Results

7.1. Fashion Retrieval

For exhaustive analysis, we experiment with multiple configurations of embeddings from GSN. An ablation study is presented in Table 1. GSN[Q] uses query trunk vector, GSN[T] uses target trunk vector, GSN[$Q \oplus T$] concatenates query and target vectors and GSN[RL] applies transformation functions learned using RL based search strategy. We discuss the specific functions learned for each of the datasets in Section 7.2.1 . To generate the target trunk volume, at inference time, the query image is propagated through the fully convolutional target trunk network (Figure 3). A $3 \times 3 \times 1024$ dim feature volume is generated and the column vectors are averaged to obtain the 1024-dim feature vector.

As discussed in Table 1, GSN[T] outperforms GSN[Q] which suggests the additional viability of our target grid as an augmentation procedure. This observation is corroborated through experiments discussed in a recent work [18]. Concatenating query and target vector representations further improves performance. Sample qualitative results are shown in Figure 6 which typify the ability of GSN framework to generate embeddings robust to large variations in pose, illumination, blur, and occlusion.

In the following subsections, comparisons are reported against WTBI [5], DARN [6], FashionNet [10], HDC [24], VAM [22] and HTL [25].

Recall	top-1	top-5	top-10	top-20	top-50
WTBI	0.03	0.04	0.05	0.06	0.08
DARN	0.03	0.06	0.08	0.11	0.15
FashionNet	0.08	0.12	0.15	0.19	0.23
VAM	0.13	0.27	0.34	0.42	0.54
GSN	**0.25**	**0.35**	**0.41**	**0.47**	**0.57**

Table 2. Recall rates comparison on Consumer-to-Shop retrieval dataset.

7.1.1 Consumer-to-Shop

Table 2 contains top-n recall rate comparison on Consumer-to-Shop (C2S) retrieval dataset. GSN outperforms existing state-of-the-art methods on recall performance. GSN outperforms [22] by 12% and [10] by 14% on top-1.

7.1.2 Inshop Retrieval

Table 3 contains top-n recall rate comparison on the Inshop retrieval benchmark. GSN outperforms all existing state-of-the-art methods, outperforming [25] by 6.7% and [24] by 24.5%. The flexibility of the GSN architecture enables integration of the adversarial hard negative mining strategy introduced in [25]. This is likely to result in further im-

[1] Shared here: https://github.com/PkuRainBow/

[2] Shared model does not match numbers mentioned in paper but can still be used as baseline to validate our experiment

[3] γ is a constant, that is equal to the number of features in the data

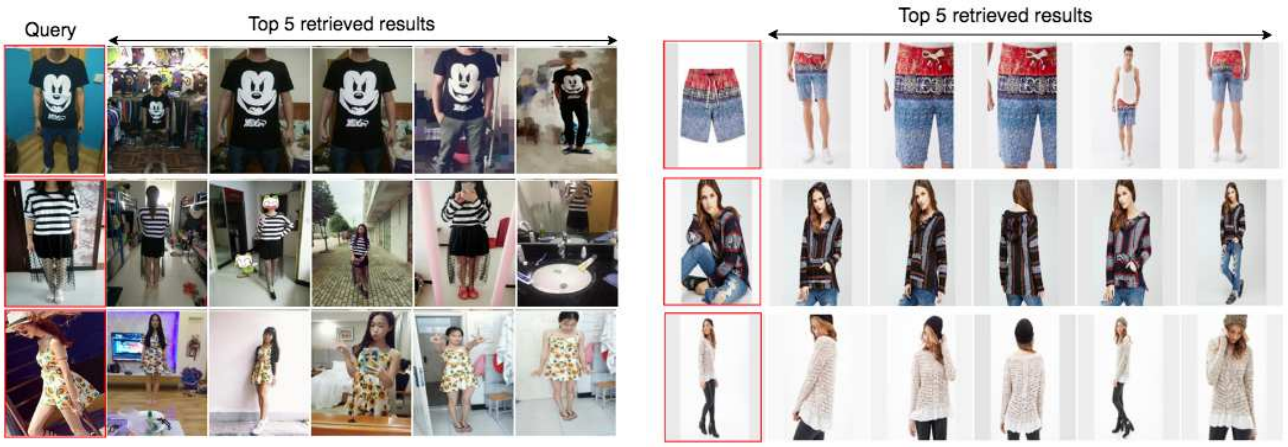


Figure 6. Sample retrieval results on Consumer-to-Shop (left) and Inshop retrieval (right) datasets. Query images are highlighted with red borders, followed by the top-5 retrieved results. Visual embeddings generated by GSN are robust to large variations in illumination, occlusion, pose, and deformation.

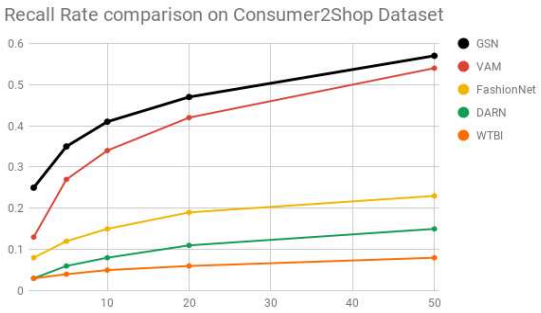


Figure 7. Recall rates for different methods under comparison on Consumer-to-shop retrieval dataset.

Recall	top-1	top-5	top-10	top-20	top-50
WTBI	0.35	0.42	0.46	0.506	0.54
DARN	0.38	0.52	0.62	0.675	0.71
FashionNet	0.53	0.68	0.73	0.764	0.8
HDC	0.62	0.85	0.87	0.89	0.93
VAM	0.67	0.84	0.89	0.92	0.94
HTL	0.803	-	0.939	0.958	-
GSN	**0.870**	**0.946**	**0.959**	**0.975**	**0.986**

Table 3. Recall rate comparison on Inshop retrieval dataset.

provement in performance against what is obtained with the current offline hard-negative sampling strategy.

7.2. Learning Feature Transformations

7.2.1 Retrieval

For our GSN architecture, the transformation function learned over the Consumer-to-Shop (Cons-2-Shop) retrieval dataset is:

$$maximum(erf(f_1),\ sqrt(f_1 + f_2)) \tag{5}$$

and over the Inshop retrieval dataset is:

$$log(1 + exp(f_1 + f_2)) \tag{6}$$

Here f_1 and f_2 refer to the feature vectors of the query and target grid trunks respectively. erf and $sqrt$ refer to the error function and square root function respectively.

Table 4 shows the results of the retrieval performance before and after the learned transformation has been applied. Note that without the learned transformation the combined representation is $concatenation(f_1, f_2)$ as in [24].

Figure 8 qualitatively validates the impact of the learned feature transformations on retrieval performance. These observations are further supported through corresponding increase in mean average precision (mAP) @0.1 from 75.2% to 76.4% on applying learned transformations to GSN embeddings for inshop dataset. mAP, as a performance metric, is better able to elucidate this impact than top-n recall as it captures the *number of positives* in the retrieved set.

To test the generalizability of our approach, we also tested it over the features obtained by [24] for the CARS [7] dataset. The learned transformation function over this dataset is:

$$tan^{-1}(concat(f_1,\ 2 * f_2,\ 3 * f_3)) \tag{7}$$

Here f_1, f_2 and f_3 are the three output feature vectors generated in the original architecture as discussed in [24].

7.2.2 Classification

We conduct our searches over FashionMNIST and MNIST classification dataset. An RNN controller emits a kernel

Figure 8. Results showing qualitative impact of learned transformations on retrieval performance for Inshop dataset. The top and bottom rows show retrieved results for GSN embeddings before and after applying learned transformations. The effectiveness of the proposed transformation strategy can be seen by the fine-grained improvement in the quality of retrieved images.

Recall	top-1		top-2		top-4	
Dataset	w/o	w/	w/o	w/	w/o	w/
CARS	0.715	**0.720**	0.816	**0.818**	0.892	**0.893**
Inshop	0.861	**0.870**	0.905	**0.913**	0.924	**0.931**
C2S	0.234	**0.247**	0.283	**0.292**	0.328	**0.337**

Table 4. Comparison of top-n retrieval performance without (w/o) and with (w/) learned transformation functions.

Kernel #samples	Val 500	Test 1000
Sigmoid	10.0	10.1
RBF	86.4	82.76
Learned Kernel	**89.0**	**87.96**

Table 5. Classification Accuracy comparison on Fashion-MNIST dataset.

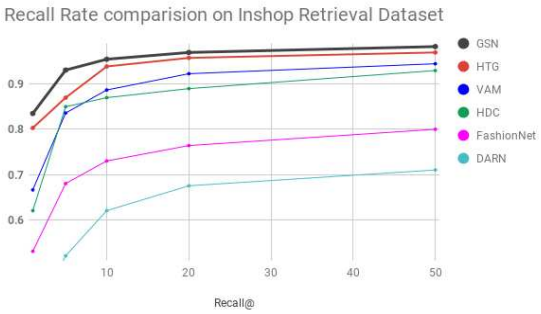


Figure 9. Recall rates for different methods under comparison on Inshop retrieval dataset.

Kernel #samples	Val 500	Test 1000	 2000
Sigmoid	81.0	74.42	84.94
RBF	86.4	87.76	87.85
Learned Kernel	**90.2**	**91.01**	**93.12**

Table 6. Classification Accuracy comparison on MNIST dataset.

function for SVM. The SVM with the emitted kernel function is trained over 1000 FashionMNIST/MNIST training samples and the accuracy over a separate 500 validation samples is used as a reward signal for the RNN controller. The final discovered kernel function for FashionMNIST is:

$$\phi_{FashionMNIST}(x, y) = ||x * y||_1 + \max(x * y, 0) \quad (8)$$

The results in Table 5 show the accuracy over 10000 FashionMNIST test samples when the SVM is trained with different kernel functions (classic kernels and discovered kernel function) over 1000 training samples.

The final discovered kernel function for MNIST is:

$$\phi_{MNIST}(x, y) = ||\min(\sin(x * y), \sin(x \cdot y/\gamma))|| \quad (9)$$

Even though the RNN controller has been trained to learn a kernel function that best fits the 1000 training samples, the discovered kernel function works better than the classic kernel functions even when it is used to fit 2000 training samples. Table 6 shows the test accuracy results when the different kernel functions are used to train SVM over 2000 samples.

8. Conclusion

We introduce a Grid Search Network (GSN) for generation of visual embeddings and highlight its effectiveness on the fashion retrieval task by achieving state-of-the-art performance on benchmark datasets. To further improve this performance, we use a reinforcement learning based strategy that learns a composite transformation function that is applied over the GSN embeddings. We emperically show that such a transformation results in boosting the retrieval performance. We reinforce claims regarding its applicability by improving performance of existing state-of-the-art methods without any additional training of the embedding network. Additionally, we extend the RL strategy to learn custom kernel functions for SVM based classification.

References

[1] I. Bello, B. Zoph, V. Vasudevan, and Q. V. Le. Neural optimizer search with reinforcement learning. *arXiv preprint arXiv:1709.07417*, 2017.

[2] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pages 3642–3649. IEEE, 2012.

[3] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[4] L. Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[5] M. Hadi Kiapour, X. Han, S. Lazebnik, A. C. Berg, and T. L. Berg. Where to buy it: Matching street clothing photos in online shops. In *Proceedings of the IEEE international conference on computer vision*, pages 3343–3351, 2015.

[6] J. Huang, R. S. Feris, Q. Chen, and S. Yan. Cross-domain image retrieval with a dual attribute-aware ranking network. In *Proceedings of the IEEE international conference on computer vision*, pages 1062–1070, 2015.

[7] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] J. Lin, O. Morère, V. Chandrasekhar, A. Veillard, and H. Goh. Deephash: Getting regularization, depth and fine-tuning right. *CoRR*, abs/1501.04711, 2015.

[10] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1096–1104, 2016.

[11] McKenzie, 2018.

[12] A.-r. Mohamed, G. E. Dahl, and G. Hinton. Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):14–22, 2012.

[13] PwC, 2018.

[14] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. 2018.

[15] F. Sadeghi, C. L. Zitnick, and A. Farhadi. Visalogy: Answering visual analogy questions. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 1882–1890, Cambridge, MA, USA, 2015. MIT Press.

[16] Shopify, 2018.

[17] R. Socher, J. Bauer, C. D. Manning, et al. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 455–465, 2013.

[18] C. Summers and M. J. Dinneen. Improved mixed-example data augmentation, 2018.

[19] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[21] F. United, 2018.

[22] Z. Wang, Y. Gu, Y. Zhang, J. Zhou, and X. Gu. Clothing retrieval with visual attention model, 2017.

[23] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[24] Y. Yuan, K. Yang, and C. Zhang. Hard-aware deeply cascaded embedding. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 814–823. IEEE, 2017.

[25] Y. Zhao, Z. Jin, G.-j. Qi, H. Lu, and X.-s. Hua. An adversarial approach to hard triplet generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–517, 2018.

[26] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.