Learning Discriminative Embeddings for Object Recognition *on-the-fly*

Miguel Lagunes-Fortiz[1,2] Dima Damen[2] and Walterio Mayol-Cuevas[2]

***Abstract*— We address the problem of learning to recognize new objects *on-the-fly* efficiently. When using CNNs, a typical approach for learning new objects is by fine-tuning the model. However, this approach relies on the assumption that the original training set is available and requires high-end computational resources for training the ever-growing dataset efficiently, which can be unfeasible for robots with limited hardware. To overcome these limitations, we propose a new architecture that: 1) Instead of predicting labels, it learns to generate discriminative and separable embeddings of an object's viewpoints by using a Supervised Triplet Loss, which is easier to implement than current smart mining techniques and the trained model can be applied to unseen objects. 2) Infers an object's identity efficiently by utilizing a lightweight classifier in the features embedding space, this keeps the inference time in the order of milliseconds and can be retrained efficiently when new objects are learned. We evaluate our approach on four real-world images datasets used for Robotics and Computer Vision applications: Amazon Robotics Challenge 2017 by MIT-Princeton, T-LESS, ToyBoX, and CORe50 datasets. Code available at [1].**

I. Introduction

Over the past few years, the robotics and computer vision community have adopted Convolutional Neural Networks (CNNs) as the standard approach for addressing object recognition and localization [2]. Despite the human-level performance in recognition accuracy, CNNs are still very limited compared with the cognition capabilities of humans [3]. One of these limitations is the ability to learn new instances efficiently, without catastrophically forgetting the previously learned ones, while considering variations in pose and appearance.

Learning new objects *on-the-fly*, that is, able to process objects as soon as they are being perceived, is a desirable capability for robots and intelligent systems working within dynamic environments [4], [5].

While continuous fine-tuning approaches have been explored [6] it is an unfeasible approach for most autonomous robotic platforms given three main limitations: 1) It assumes access to the original training set, which can easily reach the order of TBs even with low-quality images. 2) It requires high end dedicated computational resources for training the model efficiently and 3) The time required for retraining increases linearly as more items are introduced.

Miguel Lagunes-Fortiz thanks the Mexican Council of Science and Technology (CONACyT) for sponsoring his studies with the scholarship number 686450

[1] Bristol Robotics Lab, UK University of Bristol, Bristol, UK mike.lagunesfortiz@bristol.ac.uk

[2] Department of Computer Science, University of Bristol, Bristol, UK {Dima.Damen, Walterio.Mayol-Cuevas}@bristol.ac.uk

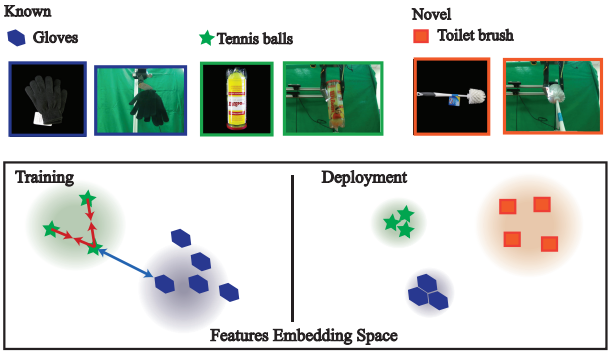


Fig. 1. The problem we aim to solve is to learn new objects without re-training the CNN model. To do so, we teach a model to maximize the separability (blue arrow) and minimize the intra-class distance (red arrows) of embeddings in a supervised manner. During deployment, the model can produce embeddings with these properties even for objects not seen during training (novel objects). In order to perform classification, we use a lightweight classifier such as Nearest Neighbours for matching embeddings with unknown and known labels.

These limitations motivate for developing models that will not require fine-tuning for learning new objects.

The general idea in our approach is to design a Deep Learning model that learns two tasks simultaneously:

1) Learns the concept of "Object similarity" by mapping object's viewpoints from the same class close to each other and dissimilar ones farther apart, in a feature embedding space $\mathbb{R}^n$.
2) Learns a powerful visual representation by combining a metric learning and fully supervised losses.

By doing so, the model learns to generate discriminative and separable embeddings generically in the features space $\mathbb{R}^n$, as depicted in Fig. 1, which then can be classified by a lightweight classifier such as Nearest Neighbours or a Support Vector Machine [7]. New objects can be mapped to $\mathbb{R}^n$ using the same learned model, and only the lightweight classifier has to be retrained in order to get the probabilities considering all learned objects.

We evaluate our approach in four datasets used for multi-view object recognition: T-LESS, ToyBox, Amazon Robotics Challenge 2017 by Princeton-MIT and CORe50. We use precision as the evaluation metric.

Contribution: The primary goal in our paper is to show that a single CNN model can be trained effortlessly to generate discriminative and separable features which can be useful for learning new objects *on-the-fly* efficiently.

II. Related work

The most straightforward strategy for learning from a continuous stream of data using Convolutional Neural Networks (CNN's) is to retrain the model (entirely or just a few layers) using the updated training set [8], [9], [6]. While these approaches offer state-of-the-art performance, it is an unfeasible approach for many autonomous and mobile robots with limited computational resources that are required to learn new objects efficiently.

An alternative to continuous fine-tuning is the Metric Learning approach, which instead of predicting labels, the model is trained to learn the concept of "similarity" by bringing close to each other embeddings from the same class and far apart otherwise, in a feature embedding space $\mathbb{R}^n$.

Currently, the best approaches to metric learning employ state of the art CNNs [10], [11]. A pioneering model is the Siamese architecture [12] which utilizes the Contrastive Loss (Eq. 1) defines as:

$$\mathscr{L}_{contrastive} = (1-Y)\frac{1}{2}D(X_1,X_2) + (Y)\frac{1}{2}\left[m - D(X_1,X_2)\right]_+ \tag{1}$$

Where X_1, X_2 are pairs of images, that can be from the same or different objects, indicated in the vector Y. A CNN f_θ with weights θ is used for mapping images to an embedding space $\mathbb{R}^n$ (Eq. 2), where the Euclidean distance (Eq. 3) d is used for computing the similarity between the two embeddings. Dot product, Mahalanobis distance or even a trainable metric as in [13], can be used as an alternative to Eq. 3. Finally, the $[\cdot]_+$ operator denotes the hinge function equivalent to $max(0,\cdot)$ function.

$$D(X_a,X_b) = d(f_\theta(X_a), f_\theta(X_b)) \tag{2}$$

$$d(x_a,x_b) = \frac{1}{2}\left\|x_a - x_b\right\|_2^2 \tag{3}$$

While the Siamese architecture has been used for One-Shot learning [14], dimensionality reduction [12], image classification [15] and cross-domain [4] with competitive results, it is unable to learn similarity and dissimilarity at the same time since the pairs are either from the same or different object.

This limitation is overcome by the Triplet architecture [16], which encourages a relative distance constraint between similar and dissimilar images simultaneously in the Triplet Loss (Eq 4).

$$\mathscr{L}_{triplet} = \left[D(X_a,X_p) - D(X_a,X_n) + m\right]_+ \tag{4}$$

The Triplet architecture became the most widely used approach for metric learning [17], with two main limitations:

1) Selecting the triplets in (Eq. 4) is a non-trivial and crucial task [16], given that f_θ quickly learns to correctly map most trivial triplets, it leaves a large fraction of all triplets uninformative and makes hard mining triplets a crucial step for training [18].
2) The loss is defined in terms of small groups of images in the mini-batch and does not consider a global structure of the training set, which might lead to sub-optimal solutions depending on the mini-batch size.

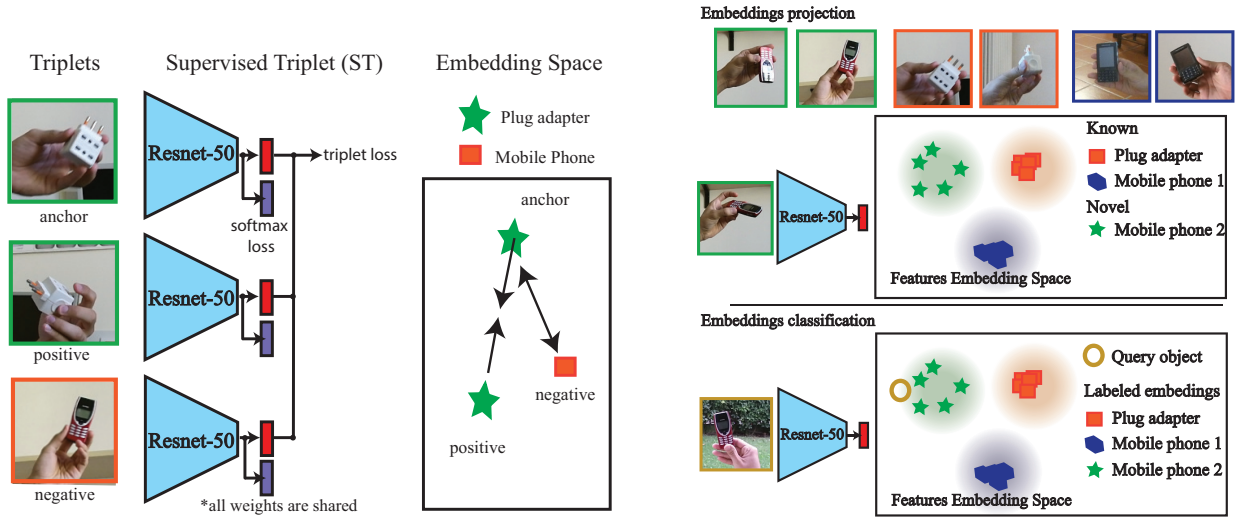
Thus, many variants of the Triplet architecture aim to address these issues. Concerning mini-batch formation, a first approach for hard mining triplets is the Lifted Structure [7] approach, which consists of incorporating on-line hard negative mining by comparing each positive example against all negative examples in the training mini-batch. The Quadruplet [19] architecture incorporates an additional negative example in the mini-batch aiming to facilitate the clustering of negative examples. Quintuple [20] and N-pair [21] models are an extension of this idea.

For selecting the best triplets, [22] proposes a smart mining trainable module that forms triplets from a pool composed of semi-hard positive and negative samples. Similarly, [23] proposes a trainable module named PDDM (Position-Dependant Deep Metric) that scores the hardest negative example within the mini-batch, based on relative and global distances within the batch. More recently, [24] proposed Batch Hard forming, where the core idea is to form batches by randomly sampling P classes (i.e., instances) and then selecting the hardest positive and the hardest negative samples within the batch. An orthogonal approach for speeding up the training is Angular Loss [25] where authors propose an additional angular loss term that constrains the upper bound angle in each triplet triangle.

For addressing the lack of global structure in the metric space, [26] proposes to combine Sample-based methods (such as Triplets or Softmax) with Set-Based methods (such SMVs). This approach is trained with a Max-Margin Loss, which improves the separability of the embeddings by maximizing the possible inter-class margin by using Support Vector Machines (SVMs). The main limitation, having to train the model set-based offline and the sample-based online.

In [15], authors propose a global loss that tries to minimize the variance within each embedding distribution and maximize the mean value of the distances between non-matching pairs. The drawback of this method is requiring the complete training set loaded in memory, in order to estimate variance and means in each class. Center Loss [27] aims for the same goal of minimizing the intra-class distance of the embeddings by learning a center for each class and penalizing the distances between the embeddings and their corresponding class centers. It works jointly with Softmax loss. Following the idea of making more discriminative embedding, [28] combines the Triplet and Center Loss for multi-view object retrieval.

The data points in the embedding space can be used for person re-identification [16], [24], clustering and retrieval [7], [21], [29], [10], [7], [28] and object recognition [14], [4] and [5]. To do so, an additional classifier has to be used. The two common approaches are using Nearest Neighbours and Support Vector Machines.



(a) Training (b) Deployment

Fig. 2. Proposed Object Recognition *on-the-fly* framework. Our Supervised Triplet Network (a) is trained with triplets of images, a soft margin triplet loss is obtained from the triplets, and a softmax loss is obtained by utilizing each image and its label. For deployment (b), the network is simplified to a single branch, since all weights are shared. Additionally, we remove the softmax layer, leaving the network to produce agnostic embeddings of 128-dimension. With the simplified network, we first project images with known labels into a common embeddings space, these images can depict objects never sought during the training of the model. For performing classification, the model will project an image's embedding close to the most similar labeled embeddings, for which a light-weight classifier, such as nearest neighbors, can be used to estimate the unknown class.

III. Proposed Method

Our approach consists of dividing the recognition problem into two stages: Embeddings generation and Classification. We use a CNN to generate separable and discriminant embeddings from an object's viewpoints in a generic fashion, so then the model can be applied to unseen objects. With separable and discriminant features projected in an embedding space, a lightweight classifier such as Nearest Neighbors or a Linear Support Vector Machine can be used for retrieving the classification probabilities efficiently. When a new object is needed to learn, only the lightweight classifier has to be re-train.

Feature generation and classification are typically done end-to-end when using CNNs [33]. However, this approach is not suitable for autonomous robots with limited computational resources, considering that it would require high memory capacity for storing the ever-growing image dataset and high computational resources for training the CNN efficiently on board. Additionally, the learned features by using full supervision only, such by a Cross Entropy Loss, are not discriminative enough since the optimization is focused only in finding decision boundaries that separate the class manifolds [28].

In the other hand, Metric Learning approaches such as the Triplet architecture [34] aims to learn a more general concept of *image similarity* that can be useful for estimating the similarity in viewpoints of novel objects. However, learning such a concept requires a significant amount of training examples (e.g. using tens of millions, compared to only millions, lead to a relative reduction error of 60% in [16]) and selecting the tuples of images in the mini-batches is still an open research field [11], which can be a difficult to address when learning from few examples.

Thus, we propose a framework that combines the benefits of both approaches by utilizing a fully supervised loss with a metric loss as regularizer as shown in Fig. 2(a). The loss function in our Supervised Triplet is defined by:

$$\mathscr{L}_{STriplet} = \mathscr{L}_{Softmax} + \lambda \cdot \mathscr{L}_{Triplet} \tag{5}$$

$$\mathscr{L}_{Triplet} = [d(f_\theta(X_a), f_\theta(X_p) - d(f_\theta(X_a), f_\theta(X_n) + m]_+ \tag{4}$$

$$\mathscr{L}_{Softmax} = -\log\left(\frac{e^{x_{class}}}{\sum_i e^{x_i}}\right) \tag{6}$$

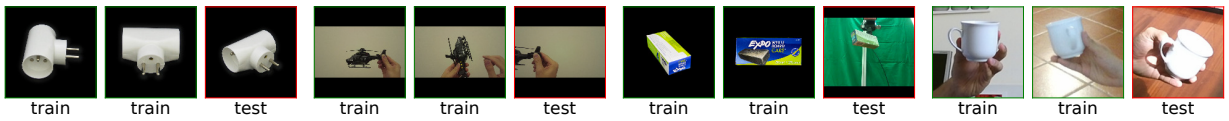
Each loss term works as follows:

A. Softmax loss

As studied in [27] and [28], using a fully supervised loss such as Softmax Cross Entropy can be beneficial for Metric Learning models. The idea behind is to use the Softmax loss for generating manifolds efficiently (e.g., by utilizing all the images in the mini-batch as oppose to the very expensive time required by triplets [10]).

B. Triplet Loss

Similar to Center Loss [27], is it possible to learn a more discriminative visual representation when the Softmax Cross Entropy loss is constrained by penalizing the similarity of the features in the Embeddings Space $\mathbb{R}^n$. We use Euclidean distance for comparing the similarity of two embeddings. As oppose to [27], we use the triplet loss, which brings embeddings from a similar class together and away otherwise without the need of computing centroids from every class, in



(a) T-LESS (b) ToyBox (c) Amazon Robotics Challenge (d) CORe50

Fig. 3. Object Recognition Datasets. In T-LESS [30], the model has to recognize unseen viewpoints. In Toybox [31] we used the hodgepodge videos for training and translations across the three axes for testing. We use ARC as in [4]. We use training and testing scenes as proposed by [32]

every training step. As studied by [28], the metric loss has to be attenuated by a factor λ in order to give priority to the manifolds generated by the Softmax Cross Entropy loss, this is the only additional hyper-parameter in our model respect to a Triplet architecture [16].

So far, the CNN model can produce cluster-like embeddings from an object's viewpoints, which can be applied to objects not seen during training. However, an additional classifier is required in order to compute the prediction probabilities considering all the objects learned. Linear Support Vector Machines and Nearest Neighbors are common choices for image retrieval, face identification and object recognition.

C. Recognizing novel objects

The methodology for projecting images into a common Features Embeddings Space and computing the prediction probabilities is the following:

1) First, the model can be compacted as shown in Fig. 2(b). Since all the weights in the model are shared, we can remove the two additional siblings and leave the model with a single branch. Additionally, we also remove the fully connected layer that was used for the Cross-Entropy Loss.
2) For learning a new object, we project the training images to the Feature Embeddings Space $\mathbb{R}^n$ and we associate each data point with the respective label.
3) We retrain the lightweight classifier for making predictions by taking into account the added data points and labels in $\mathbb{R}^n$.
4) For recognizing the object, we project a test image to $\mathbb{R}^n$, and the classifier will compute the prediction probabilities.

The training examples are stored as embeddings, so there is no need for saving image files. Any CNN architecture can be trained with Eq. 5 for producing the embeddings. Similarly, any classifier can be used for predicting the decision boundaries and probabilities in $\mathbb{R}^n$. In the next section, we describe how the selection of the classifiers affects performance and computing times.

IV. Experiments

We selected ResNet-50 [35] as the backbone CNN for generating features and K-Nearest Neighbors with $k = 5$ as the classifier in order to make our model comparable with the Amazon Robotics Challenge 2017 winner [4], who uses the same configuration. A fully connected layer generates the embeddings with a dimension of 128 elements. Selecting the embedding size has been studied by [16] and [36] in the context of face recognition and image retrieval, choosing a dimension of 128 elements leads to both faster inference and higher accuracy compared to higher dimensions.

As the first baseline, we selected ResNet-50 trained with ImageNet, for generating embeddings and K-Nearest Neighbors ($k = 5$) as the classifier. Since ResNet50 does not have intermediate fully connected layers, we follow the methodology proposed by [4] which consists in taking the feature map after the average pooling layer and flatten it to a 1-D vector with 2048 elements (this dimension resulted from using a resolution of 224x224 pixels). This baseline allowed us to explore if a model trained with a large dataset can produce discriminative embeddings for multi-view object recognition.

As the second baseline we selected the Triplet Architecture since it is the most widely used Deep Metric Learning approach [17]. We applied hard-negative mining as proposed in [16] and [7]. The hard-negative mining consisted of taking the 20% more difficult examples in the mini-batch and feeding them in the next mini-bath. The sibling CNNs were pre-trained with ImageNet as we are using a few examples per object.

For the Triplet and Supervised Triplet (our approach) we used a soft margin, this is $m = 0$ in Eq. 4, as recommended by [24]. For the Amazon Robotics Challenge dataset, we also compared our model against the winning team [4] and their baseline Siamese Network [12]. We did not select [4] as a baseline for the other datasets as it is not detailed by the authors how to select the image pairs for single or multiple (more than two) cross-domain datasets.

We selected four datasets that depict objects from an egocentric view (as would be seen from a robot's perspective in mobile robots and manipulators) and show different poses and viewpoints of every object. Apart from the Amazon Robotics Challenges by Princeton-MIT (where an actual robot took images), we selected datasets that present challenging recognition scenarios that a robot might face in real conditions.

The first dataset is T-LESS [30], which contains 30 objects with no relevant texture. Followed by ToyBox [31] a dataset depicting 360 objects manipulated by a person. Toybox allowed us to evaluate how well the model scales (we selected ToyBox over the iCubWorld dataset [37] since, at the submission time, the latter contains only 28 instances). The remaining datasets are Amazon Robotics Challenge [4] with 60 objects collected by the MIT-Princeton robot for

TABLE I

MODELS TRAINED FOR LABEL PREDICTION VS METRIC LEARNING APPROACHES

Method	TLESS	ToyBox	ARC	CORe50
CNN (Softmax)	97.31 ± 0.17	**74.96 ± 0.08**	92.31 ± 0.13	92.61 ± 0.26
CNN (ImageNet)	34.81	10.13	27.2	24.89
Triplet CNN	93.65 ± 1.49	56.98 ± 1.32	75.17 ± 1.49	74.14 ± 1.83
S-Triplet (ours)	**98.59 ± 0.42**	72.93 ± 0.54	**96.09 ± 0.21**	**94.06 ± 0.41**

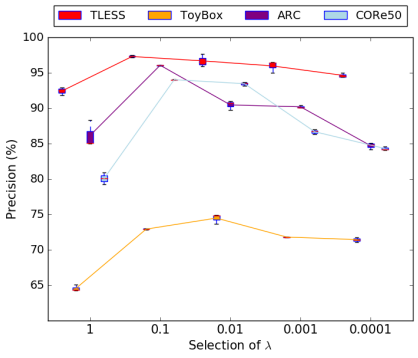


Fig. 4. For finding the best λ in Eq. 5, we varied the range from 1 to 0.0001.

TABLE II

TLESS % ACCURACY TOP-1 RECOGNITION

Method	Known	Novel	Mixed
CNN (Softmax)	**99.88 ± 0.01**	N/A	64.94 ± 1.07
CNN (ImageNet)	27.81	45.37	34.81
Triplet CNN	96.45 ± 1.58	91.03 ± 1.53	85.9 ± 1.47
S-Triplet (ours)	99.36 ± 0.23	**96.56 ± 0.25**	**95.53 ± 0.21**

TABLE III

TOYBOX % ACCURACY TOP-1 RECOGNITION

Method	Known	Novel	Mixed
CNN (Softmax)	73.94 ± 0.17	N/A	47.94 ± 0.04
CNN (ImageNet)	5.88	17.91	10.13
Triplet CNN	68.23 ± 1.72	49.6 ± 1.77	53.14 ± 1.75
S-Triplet (ours)	**81.52 ± 0.33**	**79.52 ± 0.27**	**72.23 ± 0.48**

solving the Amazon Robotics Challenge 2017 and CORe50 [32] dataset, which shows 50 objects across different environments and allows us to test the generalization capabilities not only to unseen object's poses but also new environments. Fig. 3 contains an example of training and testing conditions of every dataset. We perform the following experiments to test our approach:

A. Balancing Softmax and Metric Learning Losses:

First, we followed the methodology proposed by [28] for combining Deep Metric and Supervised Losses. The methodology consists of relaxing the triplet loss by a factor of λ, we followed their methodology of varying the value from 1e-4 to 1, in steps of one order of magnitude. We use the 80% training set for finding the best λ and evaluating on the remaining 20%. We then use the best λ with full training splits for the rest of the experiments. We show the results in Fig. 4.

B. Trade off between Deep Metric Learning and Label Prediction approaches

Here we compare the performance achieved by the models trained for label prediction with full supervision, a model train purely based in similarities (e.g., Deep Metric Learning) and our approach who combine both losses. As a baseline we chose the Triplet Architecture [34], as it is the most widely used model [7] and has higher performance than Siamese [17]. We trained the Triplet model with the hard mining methodology described in [34]. We trained all the models three times and we show the mean and standard error. Results are shown in Table I.

C. Recognizing Novel Classes:

Here we test the capabilities of each model for recognizing novels classes *on-the-fly*, this is, without retraining or fine-tuning the backbone CNN model. We followed the methodology proposed by [4] which consists in splitting each dataset into a "novel" and "split" sets, two-thirds of the classes are used for training the model and the remaining third is used for recognition of new instances. The classes are selected randomly, for the ARC dataset, we used the same splits as in [4], for TLESS, CORe50 and ToyBox the splits are located in [1]. Similarly, every model was trained three times and we show the mean and standard error accuracy. Results are shown in Tables II - V.

D. Embeddings association:

Here we explore how the performance is affected by utilizing Logistic Regression, a Linear Support Vector Machines (SVM) and K-Nearest Neighbors. Results are shown in Fig. 6.

TABLE IV

ARC % ACCURACY TOP-1 RECOGNITION

Method	Known	Novel	Mixed
CNN (Softmax)	92.31 ± 0.13	N/A	61.26 ± 0.27
CNN (ImageNet)	27.2	52.6	35.0
Siamese CNN [12]	76.9	68.2	74.12
Triplet CNN [34]	75.17 ± 1.49	58.3 ± 1.29	62.12 ± 1.57
Two-Stage K-net + N-net [4]	93.6	**77.5**	**88.6**
S-Triplet (ours)	**96.09 ± 0.21**	74.21 ± 0.26	87.53 ± 0.5

TABLE V

CORE50 % ACCURACY TOP-1 RECOGNITION

Method	Known	Novel	Mixed
CNN (Softmax)	94.22 ± 0.32	N/A	64.31 ± 0.49
CNN (ImageNet)	21.14	29.47	24.89
Triplet CNN	87.00 ± 1.49	73.13 ± 1.61	79.86 ± 1.65
S-Triplet (ours)	**95.31 ± 0.29**	**89.03 ± 0.25**	**87.23 ± 0.22**

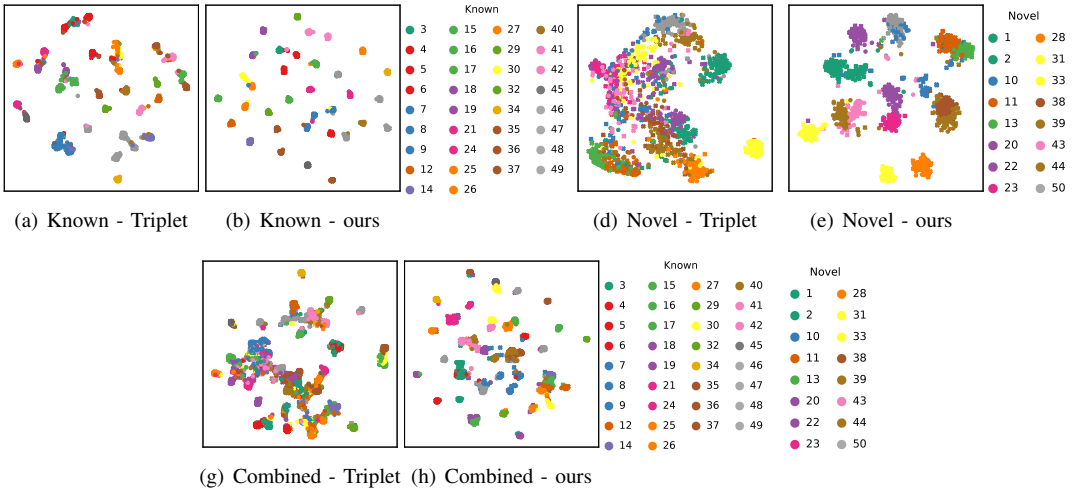


Fig. 5. T-sne visualization of the baseline Triplet Architecture [34] (left) vs. our Supervised Triplet (right) for learning novel classes in CORe50. We show the known (classes seen in training), novel (unseen object) and combined (full test set) splits and we use the labels data for coloring the clusters.

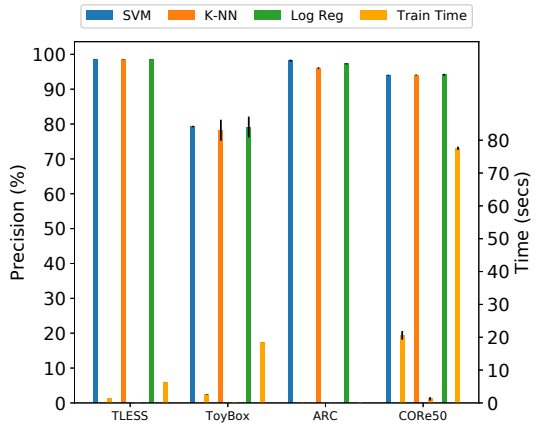


Fig. 6. We compare the performance and training time of K-Nearest Neighbours ($k = 5$), Linear SVM and Logistic Regression. We use the same training set as in Table I. We use the implementations from the scikit-learn library [38].

V. Discussion

From Table I we can see the gap in performance from a Metric Learning and Label Predictions approaches. The gap is more significant in domain-adaptation scenarios such as ARC and CORe50. This decrease in performance is an inherent drawback in current Metric Learning approaches based purely in "image similarity," since testing images depicting objects in a different background and illumination conditions are not similar enough to the training examples which lead to a wrong mapping into the correct class manifold. On the other hand, a CNN trained with a fully supervised loss dealt better with the domain adaptation scenario. Even when the model was not trained to be discriminative, the decision boundaries are flexible enough to map the testing images into the corresponding class manifold correctly.

From Table II we can conclude that all approaches were able to map correctly unseen viewpoints of each class since training and testing images have the same background and illumination conditions, the Metric Learning approach was comparable to the fully supervised. In contrast, in Table V, we show the CORe50 results, where the Metric Learning approach struggled to recognize novel objects across different environments.

In Table III we show the results with Toybox, the dataset with most objects (360). For this dataset, our approach was superior to the Metric Learning and CNN baselines, which indicates our approach is scalable since it was able to learn 120 objects and was close to the fine-tune model in Table I.

In Table IV we compare our results with the Amazon Robotics Challenge 2017 winner [4], our model has a comparable performance (only 3% less). Our model utilizes only one CNN, which translates in half of the parameters to be trained and saved. Additionally, our approach does not require a careful selection of the examples for learning the concept of "similarity" since it learns such a concept in combination with a Softmax loss, which is trained efficiently for generating manifolds. Efficient optimization means that our model does not require any cumbersome mining technique nor distance metrics, which makes implementation easier across current recognition datasets.

Finally, related to the selection of the additional classifier (Fig. 6), we found Linear SVM and Logistic regression concisely more useful that K-Nearest Neighbors with k=5, at expenses of slightly higher computational times.

In conclusion, our Supervised Triplet, had a much closer performance to the fine-tuned models across all datasets. Thus, the strategy of combining fully supervised and metric learning losses resulted in a model that generates discriminate and separable embeddings (as shown in Fig. 5) for learning new objects *on-the-fly*. Combining these losses involves choosing a factor λ which, as shown in Fig. 4, starting with a value of 0.1 and moving to nearby values resulted in being useful across all datasets.

REFERENCES

[1] M. Lagunes-Fortiz, "Pytorch implementation of supervised-triplet-network." https://github.com/MikeLagunes/Supervised-Triplet-Network, 2019.

[2] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *Behavioral and Brain Sciences*, vol. 40, p. e253, 2017.

[4] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morona, P. Q. Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, and A. Rodriguez, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2018.

[5] A. Milan, T. Pham, K. Vijay, D. Morrison, A. W. Tow, L. Liu, J. Erskine, R. Grinover, A. Gurman, T. Hunn, N. Kelly-Boxall, D. Lee, M. McTaggart, G. Rallos, A. Razjigaev, T. Rowntree, T. Shen, R. Smith, S. Wade-McCue, Z. Zhuang, C. F. Lehnert, G. Lin, I. D. Reid, P. I. Corke, and J. Leitner, "Semantic segmentation from limited training data," *CoRR*, vol. abs/1709.07665, 2017.

[6] C. Kding, E. Rodner, A. Freytag, and J. Denzler, "Fine-tuning deep neural networks in continuous learning scenarios," in *ACCV Workshop on Interpretation and Visualization of Deep Neural Nets (ACCV-WS)*, 2016.

[7] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] L. C. Jain, M. Seera, C. P. Lim, and P. Balasubramaniam, "A review of online learning in supervised neural networks," *Neural Computing and Applications*, vol. 25, pp. 491–509, Sep 2014.

[9] T. Xiao, J. Zhang, K. Yang, Y. Peng, and Z. Zhang, "Error-driven incremental learning in deep convolutional neural network for large-scale image classification," in *ACM Multimedia*, November 2014.

[10] H. O. Song, S. Jegelka, V. Rathod, and K. Murphy, "Deep metric learning via facility location," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2206–2214, July 2017.

[11] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *CoRR*, vol. abs/1306.6709, 2013.

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

[13] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning, with application to clustering with side-information," in *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, (Cambridge, MA, USA), pp. 521–528, MIT Press, 2002.

[14] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," 2015.

[15] B. G. V. Kumar, G. Carneiro, and I. D. Reid, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," *CoRR*, vol. abs/1512.09272, 2015.

[16] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pp. 815–823, 2015.

[17] X. Zhe, S. Chen, and H. Yan, "Directional statistics-based deep metric learning for image classification and retrieval," *CoRR*, vol. abs/1802.09662, 2018.

[18] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Sampling matters in deep embedding learning," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2859–2867, 2017.

[19] W. Chen, X. Chen, J. Zhang, and K. Huang, "Beyond triplet loss: a deep quadruplet network for person re-identification," *CoRR*, vol. abs/1704.01719, 2017.

[20] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5375–5384, June 2016.

[21] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems 29* (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds.), pp. 1857–1865, Curran Associates, Inc., 2016.

[22] B. G. V. Kumar, B. Harwood, G. Carneiro, I. D. Reid, and T. Drummond, "Smart mining for deep metric learning," *CoRR*, vol. abs/1704.01285, 2017.

[23] C. Huang, C. C. Loy, and X. Tang, "Local similarity-aware deep feature embedding," *CoRR*, vol. abs/1610.08904, 2016.

[24] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *CoRR*, vol. abs/1703.07737, 2017.

[25] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2612–2620, 2017.

[26] B. Gecer, V. Balntas, and T. Kim, "Learning deep convolutional embeddings for face representation using joint sample- and set-based supervision," *CoRR*, vol. abs/1708.00277, 2017.

[27] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition.," in *ECCV (7)* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), vol. 9911 of *Lecture Notes in Computer Science*, pp. 499–515, Springer, 2016.

[28] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai, "Triplet-center loss for multi-view 3d object retrieval," *CoRR*, vol. abs/1803.06189, 2018.

[29] M. Opitz, G. Waltner, H. Possegger, and H. Bischof, "Deep metric learning with BIER: boosting independent embeddings robustly," *CoRR*, vol. abs/1801.04815, 2018.

[30] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects," *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[31] X. Wang, F. M. Eliott, J. Ainooson, J. H. Palmer, and M. Kunda, "An object is worth six thousand pictures: The egocentric, manual, multi-image (emmi) dataset," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[32] V. Lomonaco and D. Maltoni, "Core50: a new dataset and benchmark for continuous object recognition," in *Proceedings of the 1st Annual Conference on Robot Learning* (S. Levine, V. Vanhoucke, and K. Goldberg, eds.), vol. 78 of *Proceedings of Machine Learning Research*, pp. 17–26, PMLR, 13–15 Nov 2017.

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[34] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, June 2014.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, IEEE Computer Society, 2016.

[36] D. P. Vassileios Balntas, Edgar Riba and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," in *Proceedings of the British Machine Vision Conference (BMVC)* (E. R. H. Richard C. Wilson and W. A. P. Smith, eds.), pp. 119.1–119.11, BMVA Press, September 2016.

[37] G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, and L. Natale, "Teaching icub to recognize objects using deep convolutional neural networks," 2015.

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.