Centroids Triplet Network and Temporally-Consistent Embeddings for In-Situ Object Recognition

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

***Abstract*— This work proposes learning to recognize objects from a small number of training examples collected and deployed *in-situ*. That is, from data collected where the objects are commonly placed or being used, perhaps after first encountering them, the learning algorithm immediately is able to recognize them again. We refer to this methodology as in-situ learning, and it opposes to the conventional methodology of using complex data acquisition mechanisms, such as rotating tables or synthetic data, to build a large-scale dataset for training convolutional neural networks (ConvNets). To learn in-situ, we propose a novel loss function that generates discriminative features for known and unseen objects, by utilizing a regularization term that reduces the distance between features and their manifold centroid. Additionally, we propose a temporal filter that is particularly useful to quickly react to appearing objects on the scene, which depending on the distance between neighboring video-frame features, it applies a weighted average between the current and the previous frame. Our framework achieves *state-of-the-art* accuracy for *in-situ* and *on-the-fly* learning, for the case of known objects achieves an average increase in accuracy of 3.01%, an increase of 3.3% for novel objects, and an average increase of 7.07% for the combined case, compared with the closest baseline. Utilizing the temporal filtering, led to a further increase in accuracy against nuisances of 7.32% for the known and novels objects case.**

I. Introduction

As robots are used for performing tasks under unstructured and dynamic environments (Fig. 1), there is a need for them to have a high level of autonomy, this is, without requiring human intervention when they are performing their duties and when they face a novel problem to address at hand.

To increase the level of autonomy in robots, we propose an approach that gives them the capability of learning to recognize specific objects (*instances*) without the need of recurring to complex image-acquisition systems or generating synthetic examples. Instead, they utilize a small amount of training examples, *i.e.* low hundreds of training examples per instance, as contained in a few seconds of video data. Since our algorithms are meant to be used by robots and other autonomous platforms, we achieve real-time rates during deployment, *e.g.* in the magnitude of dozens of frames per second, and without requiring an enormous amount of computational resources such as multiple GPUs.

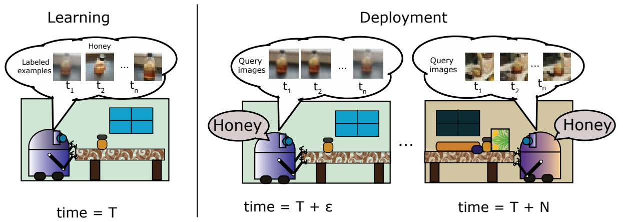


Fig. 1. An agent learning to recognize a honey bottle from a few amount of data collected *in-situ*, with scalable, robust, real-time performance, and onboard learning capabilities. Learning is performed at T and deployed within seconds ($T+\epsilon$), and the model should generalize to an unseen condition such as changing illumination, clutter, and occlusions subsequently encountered at ($T+N$). Best viewed in digital version.

We aim for a scalable real-time recognition system that can process *in-situ* data, as we argue that is a more straightforward approach of data collection compared to using complex data-acquisition setups, such as rotating tables, for obtaining pictures depicting them with ideal imaging conditions (*in-vitro* images). Additionally, we aim for a robust recognition system that can deal with the commonly encountered changes in illumination, perspective, scale, backgrounds, and occlusions.

The main challenges associated with in-situ learning includes recognizing novel objects efficiently, considering the constrained computational resources on-board for most robotic platforms, as well as the unavoidable domain shift, present when training a model with data collected in one environment and deploying in unseen conditions.

Our strategy to address these challenges consists of designing a discriminative model that can associate images depicting the same object, even when they belong to different environments. To do so, we propose the Centroids Triplet Network (CTN) for minimizing the distance between embeddings from the same objects, but also minimizing the distance with respect to their centroids (*prototypes*), while maximizing the separation to the closest centroid from another object's manifold. At inference, we propose utilizing the nearest centroids algorithm, to keep scalability and real-time performance at deployment. To deal with nuisances and ambiguous viewpoints, we propose the use of a temporal filter that enforces the temporal coherence that exists in video-data, which prevents sudden changes in the predictions between neighboring video-frames.

We evaluate our approach on datasets tailored for in-situ object recognition and compare it against *state-of-the-art* methods for learning discriminative features and generating

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 malfkov@gmail.com

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

stable predictions in ConvNets.

II. Literature Review

In-situ recognition of objects, understanding this as objects depicted within their natural or common environments, as opposed to using *in-vitro* pictures, was first addressed by [1], [2]. In [1], the authors focus on achieving domain generalization from *in-vitro* to *in-situ* data, while [2] combines features from *in-vitro* datasets with features extracted *in-situ* to build a large-scale real-time recognition system. As limitations, [1] requires *in-vitro* examples to extract clean descriptors and concluding that collecting *in-situ* data for training would be an impractical practice. On the other hand, [2] requires that the class of the desired instance to be learned is present in the ImageNet dataset to build a robust classifier.

More recently, the Amazon Robotics Challenge 2017 presented the new requirement of learning novel objects efficiently by providing a set of unseen objects two hours before the competition. Since Amazon provided *in-vitro* images from such objects, winning teams [3], [4] proposed metric learning techniques to achieve domain adaptation between the images captured by the robot *in-situ* and the provided *in-vitro* images.

While [3], [4] were designed for domain adaptation between *in-vitro* and *in-situ*, they empirically demonstrated that a ConvNet can be used to learn new objects without having to retrain the model by performing the k-nearest neighbors search in the features space, where data points from the same object are close to each other and separated otherwise. [5] builds on the same idea of utilizing a discriminative ConvNet and simplifies the model into a single branch that uses a combination of Softmax and Triplet Loss for achieving *state-of-the-art* accuracy for learning objects *on-the-fly*. We follow this research direction of utilizing discriminative networks for learning new objects without the need for retraining, and therefore, we focus this literature review on supervised approaches that learn discriminative features by utilizing regularizers in the loss function.

Regardless of the architecture design, it is now broadly studied that the commonly used combination of Cross-Entropy Loss and the Softmax function in the last fully connected layer, a.k.a. *Softmax Loss*, does not explicitly optimize the feature embedding to enforce higher similarity for intra-class samples and diversity for inter-class samples [5], [6], [7], [8], [9], [10], [11], [12], [13].

In this regard, the main approach for learning discriminable features consists of combining the Softmax Loss with regularizers that enforce the intra-class clustering and inter-class separation. These regularizes can be divided into Euclidean regularizers and angular-margin loss functions.

State-of-the-art Euclidean regularizers include Center Loss [9], where the ConvNet learns centers and the clustering of data points around those ones; Triplet Center Loss [14], originally proposed for 3D object retrieval, proposes the incorporation of the Triplet Loss with Center Loss in order to enforce inter-class separability of clusters; Similarly, [5] combines the Triplet Loss with the Softmax Loss where the features for each task are separated into two different heads, in order to improve the accuracy for classifying novel objects. In all these approaches, a hyper parameter is used to balance the supervision signals.

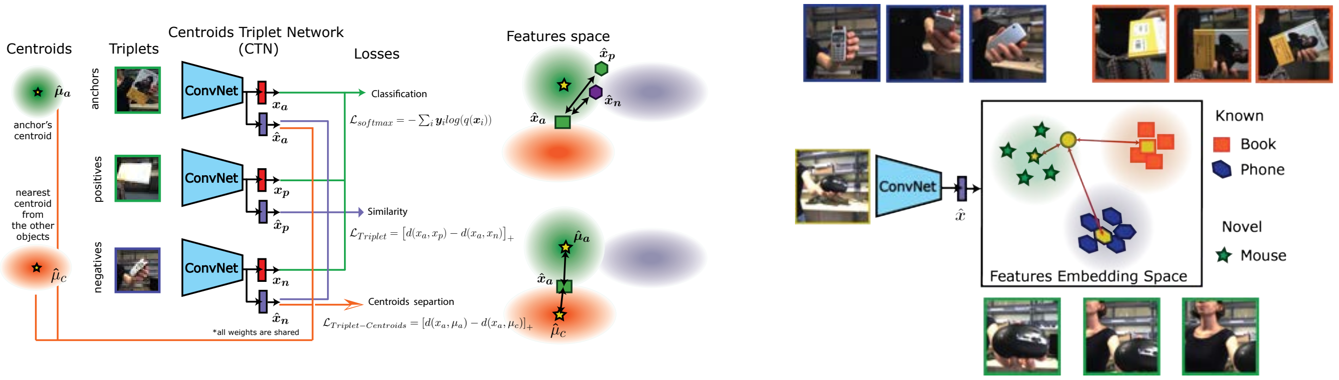
On the other hand, CosFace [11], SphereFace [10] and ArcFace [12] posit as the *state-of-the-art* angular-margin approaches and build on the findings from Large-Margin Softmax (L-Softmax) [8], which proposes a margin in the cosine product between the weights w and features x in the fully connected layer used for classification. As their name suggests, these approaches come from the facial recognition community and consist of angular constraints applied into the cosine version of the Softmax Loss. Additionally, for performing person identification, authors utilize cosine similarity to compare a query feature, against the features in the database.

Producing stable predictions is also the focus of this work. Stability Training [7] and Single-Frame Regularization [13] are the closest approaches to our goal. In Stability Training, authors propose reducing the dissimilarity in the embeddings between an image X and a variant of it with a small perturbation $T(X) = X + \Delta X$, where the perturbation ΔX is described as per-pixel independent normal distributed noise $\Delta X \sim N(0, \sum)$, with $\sum = \sigma^2 I$. In [13], where the goal is to achieve consistency for image-to-image translation, the authors propose reducing the Euclidean between embeddings produced by an image X, where an affine transformation T has been applied before and after the translation: $\mathcal{L}_{trans-inv} = \|f(T(X)) - T(f(X))\|_2$.

As studied by [13], methods for enforcing temporal consistency in image processing are mostly based on estimating dense motion, optical flow, or using recurrent neural networks. While these approaches have shown usefulness, they all suffer from one or more of the following problems: 1) Training a ConvNet and a RNN are commonly separated training stages since one requires shuffled examples and the other sequential ones. 2) high complexity and application-specific architectural modifications, 3) a significant increase in computational complexity for training and inference, 4) failure in situations where motion estimation is difficult, such as image regions with occlusion or lack of texture. Since these limitations make the approaches above unsuitable for *in-situ* learning, we aim to achieve robustness using an external temporal filter that uses the embeddings produced by the ConvNet.

III. Proposed Method

Our work builds on the findings from [5] for learning to recognize objects *on-the-fly*. However, to make the scalable real-time recognition system that we are after, we propose the following contributions. First, we replace the common and costly k-nearest neighbors search for inference used by [3], [5] for the normalized nearest centroids algorithm in the features space. Secondly, we propose an additional distance constraint that enforces discriminability between embeddings and their instance centroids, while maximizing the separation against the closest centroid from another object. Third, to



(a) Training

(b) Inference

Fig. 2. Proposed object recognition framework for *in-situ* learning. Our Centroid Triplet Network (a). Classification is performed by the nearest centroids search in the features space (b).

exploit the temporal consistency in the video frames, we propose a temporal filter that takes into account predictions from previous frames as well as the distance between them. We now explain each of these contributions:

A. Learning Discriminative Features on-the-fly

Since we are performing the nearest centroids search during deployment, we first propose a regularization term for minimizing the distance between a given embedding x_a, its corresponding instance centroid μ_a and maximizing its separation to the the closest centroid μ_c from another object, as shown in the following equation:

$$\mathcal{L}_{Triplet-Centroids} = [d(x_a, \mu_a) - d(x_a, \mu_c)]_+ \quad (1)$$

This formulation is fundamentally different from the Stability Training [7] and Single-Frame Regularization [13] approaches, where artificial perturbations are obtained by injecting Gaussian noise X', as well as applying affine transformations T into a given image X, and then minimizing the euclidean distance between the embeddings. It is also different and more efficient than directly minimizing the loss with respect to their centroids as in Deep Vector Quatization [6], and shown in section IV.

The loss function that we propose to learn discriminative features is the following:

$$\mathcal{L}_{CTN} = \mathcal{L}_{Classification} + \alpha \cdot \mathcal{L}_{Triplet-Centroids} + \beta \cdot \mathcal{L}_{Similarity} \quad (2)$$

$$\mathcal{L}_{Classification} = \mathcal{L}_{Softmax} = -\sum_i y_i log(q(x_i)) \quad (3)$$

$$\mathcal{L}_{Triplet-Centroids} = [d(x_a, \mu_a) - d(x_a, \mu_c)]_+ \quad (4)$$

$$\mathcal{L}_{Similarity} = \mathcal{L}_{Triplet} = [d(x_a, x_p) - d(x_a, x_n)]_+ \quad (5)$$

The model performs image classification utilizing the features x from the dense layer in red in Fig. 2(b). The first regularizer utilizes the features x_a from the anchor images and the distance between its class centroid μ_a, and the closest centroid μ_c from the other objects; this regularizer enforces discriminability between features and centroids, useful when the features are not distributed as spherical Gaussian due to the limited training data conditions. The second regularizer consists of the Triplet Loss, which utilizes features x_a, x_p, x_n from the denser layer in blue, and enforces discriminability between features from the same and different objects.

Each regularizer is multiplied by a hyper parameter, α and β respectively in 2, which controls the contribution of each regularizer and their values are found empirically, as discussed in the ablation studies.

To train our approach efficiently, we compute all the centroids sparsely every number of epochs n, as oppose to every mini-batch iteration. Furthermore, we utilize only the closest centroid per example to compare, as oppose to compare each embedding against all other centroids (Fig. 2).

B. Accelerating Inference Time

During deployment, we use the embeddings produced by the dense layer in blue from Fig. 2(b). We first compute the centroids $\hat{\mu}_l$ by utilizing the labeled examples $(\hat{x}_1, y_1), (\hat{x}_2, y_2), ...(\hat{x}_n, y_n)$. We then sum all the embeddings belonging to the same object $l \in Y$, and then normalize the resulting vector as indicated in the equation:

$$\hat{\mu}_l = \frac{1}{\|\sum_i (\hat{x}_i)\|} \sum_i (\hat{x}_i) \quad (6)$$

To estimate the identity $\hat{y}$ from a given embedding $\hat{x}$, of a query image, we selected the identity of the closest centroid $\hat{\mu}$ found:

$$\hat{y} = argmin_{l \in Y} \|\hat{\mu}_l - \hat{x}\| \quad (7)$$

C. Temporally-Consistent Embeddings

As studied in [7], unstable learners can classify neighboring video-frames inconsistently due to visual perturbations such as noise. In this work, we aim to empirically demonstrate that in these unstable classifiers, there is a direct relation between inconsistent predictions and dissimilar embeddings. Furthermore, by enforcing the similarity between neighboring video-frame embeddings, it is possible to make ConvNets more robust against such nuisances.

To obtain temporally-consistent embeddings, we then propose a temporal filter that performs a weighted average between the current and previous embeddings. The temporal filter is applied if the Euclidean distance d between the current and previous video-frame embeddings is lower than a threshold δ. A *big* Euclidean distance d suggests a different object on the scene, and therefore, we give priority to the current prediction. We describe the temporal filter in the following equation:

$$\hat{y}_t = \begin{cases} argmin_{l \in Y} \left\| \hat{\mu}_l - \hat{x}_t \right\|, & \text{if } d \geq \delta \\ (1-\gamma) \cdot argmin_{l \in Y} \left\| \hat{\mu}_l - \hat{x}_t \right\| + \gamma \cdot \hat{y}_{t-1}, & \text{o/w} \end{cases} \tag{8}$$

The hyper parameter γ is the weighting factor between the current and previous frame prediction, the values γ and δ are to be found empirically, and d is the Euclidean distance between the current and previous video-frame embeddings.

IV. Experimental Results

A. Datasets

We selected four datasets that depict objects from an egocentric view, as would be seen from a robot's perspective for most mobile robots and manipulators. Apart from the iCub dataset, where an actual robot took images, we selected datasets that present recognition scenarios that emulate the *in-situ* learning scenario that we are after.

To test the models capability to learn additional objects *on-the-fly*, we utilize the methodology proposed by [3], which aims to measure the *discriminability* of the features generated from seen and unseen objects. To do so, each dataset is divided into **novel** and **known** sets, where two-thirds of the objects composed the **known** set which is used for training the model and the remaining third correspond to the **novel** set which is used for recognizing new instances without fine-tuning the model, and performing the nearest centroids search in the embeddings space. Each dataset is described as follows:

CORe50 [15]: Originally proposed for continuous learning, this dataset shows 50 objects across eleven environments and allows us to test the generalization capabilities not only to unseen objects' poses but also to new environments. We utilize the standard testing set composed by scene 3, 7 and 11, and only scene 1 is used for training the model.

ToyBox [16]: It is composed of 360 toys manipulated by a demonstrator. Toybox allowed us to evaluate how well the model scales, by learning 120 novel objects *on-the-fly*. We utilize the hodgepodge videos for training and the translations and rotations across x,y,z-axis for testing. The testing set also depicts new conditions such as changes in scale, partial views, and occlusions.

iCub transformations [17]: This dataset contains 200 household objects shown by a demonstrator to an iCub Robot. We arbitrarily selected the mixed manipulations set, taken with the left camera for training the model and mixed manipulations set but the following day for testing. The testing set depicts additional backgrounds, viewpoints, and scales.

In-situ household: Since none of the *state-of-the-art* datasets allow evaluating the more realistic situation where the training images are collected within the place where such items are commonly used, such as TV remotes in a living room, we propose a new dataset to assist the benchmarking of *in-situ* learning approaches by depicting each instance in its commonplace. Our dataset consists of 20 objects with deferring training and testing conditions, without hand presence but depicting a variety of viewpoints, scales, clutter, occlusions, and illumination conditions. We plan to add more places and objects in the future.

B. Baselines

To evaluate the effectiveness of our method, we selected *state-of-the-art* approaches for learning discriminative features, as it is relevant when learning objects *on-the-fly*, as well as approaches that aim for robust and temporally consistent predictions without the use of recurrent connections. All the backbone ConvNet in the following models consist of ResNet-50, with an additional Dense Layer with a dimension of 1024 elements which is used as an embedding. We initialized all the models with weights learned from ImageNet, and use stochastic gradient descent as the optimizer with a learning rate of $l_r = 1 \times 10^{-3}$ and momentum $\mu = 0.9$. Each model was trained three times, and we chose the checkpoint with the best performance for the combined case of recognizing known and novel objects. We selected the following hyper parameters for each baseline, each of them are explained in their corresponding citation:

Deep Vector Quantization [6]: As a first experimental baseline we utilize a loss function that minimizes the Euclidean distance between features and its class manifold centroid, given by $\mathcal{L} = \|x - \mu\|_2$.

Stability Training [7]: We utilize $\alpha = 0.01$, and for generating random noise, we use a standard deviation of $\sigma = 0.04$.

Invariance Regularization [13]: We utilize $\alpha = 0.95$ and the affine transformations described in [13]. We use the same affine transformations as data augmentation for the other models, in order to make all approaches comparable.

CenterLoss [9]: We selected the hyper parameters $\lambda = 0.1$ and $\alpha = 0.005$ in the loss function [9] and utilized the PyTorch implementation from [18].

S-Triplet [5]: We selected the hyper parameters $\lambda = 0.0001$ in the loss function [9] and utilized the PyTorch implementation from [19].

Angular-margin approaches: **CosFace** [11], **SphereFace** [10] and **ArcFace** [12] posit as the state-of-the-art angular-margin approaches for learning discriminative features. We

TABLE I

RECOGNIZING KNOWN OBJECTS % ACCURACY TOP-1 RECOGNITION

	Core50	ToyBox	iCub	in-situ household	Average
Deep Vector Quatization [6]	53.40 ± 0.45	62.13 ± 0.91	82.21 ± 1.72	53.21 ± 1.12	62.98 ± 1.02
Softmax Loss	69.50 ± 0.95	71.73 ± 0.91	93.57 ± 1.11	64.95 ± 1.12	74.93 ± 1.02
Stability Training [7]	71.73 ± 0.87	71.91 ± 0.85	90.53 ± 0.85	79.14 ± 1.15	78.33 ± 0.93
Invariance Regularization [13]	74.86 ± 0.7	72.52 ± 0.98	92.81 ± 0.98	**82.05 ± 1.17**	80.56 ± 0.95
ArcFace [12]	63.30 ± 1.01	85.91 ± 1.32	92.28 ± 1.34	66.18 ± 1.13	76.91 ± 1.2
Center Loss [9]	57.58 ± 0.95	76.86 ± 1.11	89.19 ± 1.07	78.02 ± 1.13	75.41 ± 1.07
S-Triplet [5]	74.47 ± 1.15	80.26 ± 1.12	92.68 ± 1.11	77.39 ± 1.19	81.21 ± 1.14
CTN (ours)	**75.53 ± 1.06**	**86.13 ± 1.03**	**94.42 ± 1.01**	80.82 ± 1.16	**84.23 ± 1.18**

TABLE II

RECOGNIZING NOVEL OBJECTS *on-the-fly* % ACCURACY TOP-1 RECOGNITION

	Core50	ToyBox	iCub	in-situ household	Average
Deep Vector Quatization [6]	41.22 ± 0.95	51.73 ± 0.91	63.57 ± 1.11	64.95 ± 1.12	55.36 ± 1.02
Softmax Loss	59.93 ± 1.13	69.73 ± 0.95	79.18 ± 1.03	72.98 ± 1.29	70.46 ± 1.01
Stability Training [7]	60.13 ± 0.98	75.02 ± 0.72	79.09 ± 1.02	80.20 ± 1.25	73.61 ± 0.99
Invariance Regularization [13]	56.92 ± 0.87	73.77 ± 0.91	80.37 ± 1.09	79.31 ± 1.15	72.60 ± 1.00
ArcFace [12]	61.93 ± 1.11	80.89 ± 1.28	77.40 ± 1.13	**94.76** ± 1.02	78.75 ± 1.13
Center Loss [9]	57.58 ± 1.01	51.96 ± 1.09	49.87 ± 1.04	81.30 ± 1.11	60.18 ± 1.06
S-Triplet [5]	64.67 ± 1.15	77.30 ± 1.05	82.36 ± 1.11	90.40 ± 1.51	79.18 ± 1.21
CTN (ours)	**70.11 ± 1.09**	**85.23 ± 1.05**	**87.11 ± 1.64**	87.49 ± 1.21	**82.48 ± 1.18**

TABLE III

RECOGNIZING KNOWN AND NOVEL OBJECTS *on-the-fly* % ACCURACY TOP-1 RECOGNITION

	Core50	ToyBox	iCub	in-situ household	Average
Deep Vector Quatization [6]	39.34 ± 0.95	55.32 ± 0.92	62.17 ± 1.11	44.23 ± 1.32	50.26 ± 1.02
Softmax Loss	52.84 ± 1.14	62.73 ± 0.85	81.66 ± 1.02	50.14 ± 1.19	61.84 ± 1.05
Stability Training [7]	53.73 ± 1.19	71.96 ± 0.98	79.85 ± 1.02	71.96 ± 1.13	69.38 ± 1.08
Invariance Regularization [13]	52.62 ± 1.05	72.52 ± 0.93	82.05 ± 1.11	72.52 ± 1.15	69.93 ± 1.06
ArcFace [12]	50.71 ± 1.04	79.65 ± 1.21	79.96 ± 1.35	64.16 ± 1.01	68.62 ± 1.15
Center Loss [9]	44.92 ± 1.21	59.76 ± 1.08	63.30 ± 1.01	58.80 ± 1.12	56.69 ± 1.10
S-Triplet [5]	55.61 ± 1.05	78.26 ± 1.12	82.82 ± 1.02	64.37 ± 1.32	70.26 ± 1.13
CTN ours	**60.21 ± 1.14**	**85.55 ± 1.02**	**87.32 ± 1.16**	**76.25 ± 0.97**	**77.33 ± 1.09**

TABLE IV

RECOGNIZING KNOWN AND NOVEL OBJECTS *on-the-fly* WITH TEMPORAL FILTERING % ACCURACY TOP-1 RECOGNITION

	Core50 $\alpha = 0.95, \delta = 4$	ToyBox $\alpha = 0.95, \delta = 4$	iCub $\alpha = 0.95, \delta = 5$	in-situ household $\alpha = 0.98, \delta = 4$	Average
Deep Vector Quatization [6]	45.21 ± 0.95	59.21 ± 0.91	67.12 ± 1.11	49.95 ± 1.12	55.39 ± 1.02
Softmax Loss	59.19 ± 1.14	64.37 ± 0.85	85.21 ± 1.02	55.14 ± 1.19	65.94 ± 1.05
Stability Training [7]	66.52 ± 1.19	76.27 ± 0.98	84.31 ± 1.02	77.01 ± 1.13	76.03 ± 1.08
Invariance Regularization [13]	66.81 ± 1.05	78.10 ± 0.93	86.05 ± 1.11	79.29 ± 1.15	77.56 ± 1.06
ArcFace [12]	62.33 ± 1.04	82.64 ± 1.21	86.22 ± 1.35	69.13 ± 1.01	75.08 ± 1.15
Center Loss [9]	61.47 ± 1.21	72.32 ± 1.08	68.30 ± 1.01	65.21 ± 1.12	66.82 ± 1.10
S-Triplet [5]	67.51 ± 1.05	83.12 ± 1.12	88.74 ± 1.02	77.98 ± 1.32	79.48 ± 1.13
CTN ours	**71.05 ± 1.14**	**90.01 ± 1.02**	**92.30 ± 1.16**	**85.25 ± 0.97**	**84.65 ± 1.09**

utilize the generalized loss function and implementation presented in ArcFace [12], where the parameters m_1, m_2, and m_3 represent the constraint proposed in CosFace, SphereFace and ArcFace, respectively. As hyper parameters, we selected $m_1 = 0.35$, $m_2 = 0.5$, $m_3 = 4$, and $s = 30$. As in their original implementation, we utilize cosine distance to measure the similarity between embeddings.

Softmax Loss: We utilize the most commonly used loss function for classification, consisting of a combination of Cross-Entropy loss with a softmax operation.

C. Implementation Details

We initialize the backbone ConvNet from a pre-trained model with Imagenet. Thus, our model works with RGB images with size 224 × 224. For training our model, we use mini-batches of 64 images, we use stochastic gradient descent with a learning rate of $l_r = 1 \times 10^{-3}$, momentum $\mu = 0.9$ and weight decay regularization of $wd = 1 \times 10^{-4}$. Code with training scripts, links to download our dataset, and details on the instances chosen as known and novel can be found in here.

D. Recognizing Known and Novel Objects on-the-fly

To get an understanding of the usefulness of our proposed model, we show in Tables I - III the accuracy and standard error for three different situations: (a) Only using known objects, (b) only using novel objects and (c), a general case where there is no assumption about the object to test, and embeddings of known and novel objects are used for estimating the identity of a query object.

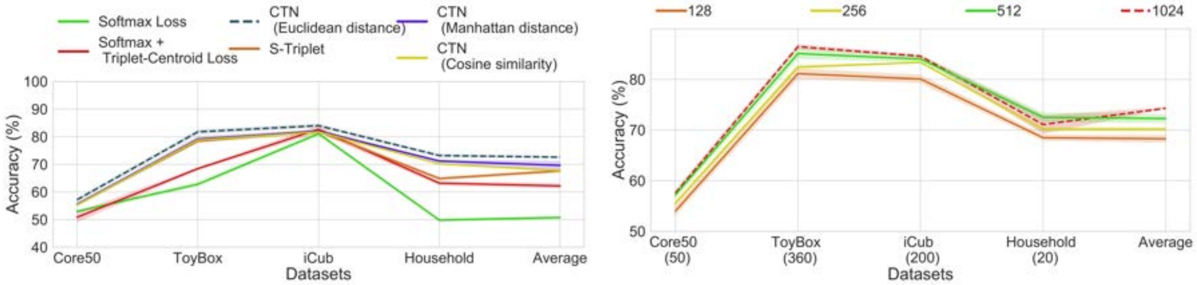
Overall, for the case of known objects, our model achieved an average increase in accuracy of 3.01% compared against the closest baseline, the S-Triplet model. For the case of novel objects, it achieved an average increase of 3.3 %, and an average increase of 7.07% for the general case.

The temporal filter that uses the embeddings distances was consistently useful for increasing the accuracy for all the approaches, as we show in Table IV, with the corresponding γ and δ values used. For our model, there was an average increase in accuracy of 8.33% for the general case of known and novel objects combined. With the model still making mistakes when there are ambiguous viewpoints from the very beginning of a testing sequence, especially confusing novel objects with known ones.

E. Scalability and Real-Time Inference

To test scalability, we considered the general case of recognizing *known* and *novel* objects (Table III). We compare the accuracy, the wall-clock time taken for evaluating all testing images and the storage required for saving all training embeddings and their instance centroids.

We noticed a consistent decrease of around 2% in accuracy in each dataset of our normalized-nearest centroids algorithm against k-nearest neighbors ($k = 5$). However, there was a considerable reduction in storage required and inference



(a) Losses and distance metrics

(b) Embeddings size

Fig. 3. Ablation Studies

time. For all the datasets, storing only the centroids represented less than 1% of storage required compared to storing all training embeddings as required in k-nearest neighbors. Related to the inference time, the nearest centers algorithm performed two orders of magnitude faster than k-nearest neighbors (with $k = 5$), making it an overall more suitable approach for a *scalable* and *real-time* object recognition system.

F. Hyper parameters Searching

In order to find a suitable value for the controlling α and β hyper parameters in 2, we performed a grid search, ranging values from 1×10^{-1} to 1×10^{-5} by decreasing an order of magnitude each step for both hyper parameters. We started by finding the best value for α by setting $\beta = 0$. With the best overall value of $\alpha = 1 \times 10^{-1}$, we found $\beta = 1 \times 10^{-2}$ to be the best overall value. In general, α and β with values greater than 1×10^{-1} causes a degradation in accuracy. Therefore, we recommend choosing $\alpha = 1 \times 10^{-1}$ and $\beta = 1 \times 10^{-2}$ as starting points.

Similarly, for the temporal filter we explore the values of α from 0.50 to 0.99 in increments of 0.01 and keeping $\delta = 3$. Once we found the best value, which is around 0.95, we then explore values of delta from 1 to 6 in increments of 0.5. We found values around 4.5 to be the most suitable in our datasets.

G. Ablation Studies

As ablation studies, we first explore similarity metrics for comparing embeddings from neighboring video-frames. We considered Manhattan distance, cosine similarity, and Euclidean distance. To evaluate the usefulness of the additional Triplet-Centroids loss, we compare our proposed model against the S-Triplet, Softmax Loss, and a Supervised Centroids loss. We found that using Euclidean distance resulted in the highest overall accuracy, as shown in Fig. 3(a). We also explored different sizes for the embeddings, as we show in Fig. 3(b). Bigger sizes achieved an overall higher accuracy, with an approximate increase of 2% by doubling the embedding dimension. Noticeably, for the household dataset, there was a decrease in accuracy when using a dimension of 1024, indicating that the regularization offered by our loss function and weight decay are not enough to compensate the lack of training data and the model starts overfitting.

H. Discussion

As shown in Table III, using the proposed Centroids Triplet Loss led to an average increase of 7.07% in accuracy, compared to the S-Triplet approach. This increase in accuracy suggests that, with the in-situ datasets used, the features produced by the S-Triplet did not distribute as a *Gaussian hyper-sphere* around its centroid, meaning that there were features closer to other objects' centroid than its corresponding one, this was more notorious in the proposed Household and iCub dataset. In this regard, there is still more to be known about the properties of the manifolds generated in the *in-situ* learning scenario and we leave that as a future research direction. Replacing the k-nearest neighbors search by the nearest centroids algorithm resulted in a highly beneficial approach for the onboard learning capabilities that we are after, since using this algorithm allowed a faster inference time by up-to two orders of magnitude, and a storage space of only 1.8 KB of memory per object, using an embedding size of 1024 elements with numpy 1.18.1 and Python 3.7.1.

With respect to the proposed weighted average used as a temporal filter, while adding the embeddings separation helped to quickly react when new objects appear on the scene and gaining a further average increase of 7% in accuracy, there are cases where using a static threshold δ was not sufficient.

In conclusion, the combination of a classification, image similarity, and centroids-discriminability losses allowed our Centroids Triplet Network to learn generalizable features for in-situ object recognition. The use of the nearest centroids algorithm contributed to maintaining scalability and real-time performance when learning additional objects. Finally, the temporal filter was a helpful strategy for increasing the network robustness for known and novel objects.

Acknowledgments

We would like to thank the visual egocentric research activity partially funded by UK EPSRC EP/N013964/1. We also want to thank Google for the GCP credits.

References

[1] M. Merler, G. C, and B. S, "Recognizing groceries in situ using in vitro training data," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007.

[2] D. Göhring, H. Judy, R. Erik, S. Kate, and D. Trevor, "Interactive adaptation of real-time object detectors," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1282–1289, 2014.

[3] 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.

[4] 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.

[5] M. Lagunes-Fortiz, D. Damen, and W. Mayol, "Learning discriminative embeddings for object reconition on-the-fly," in *ICRA*, 2018.

[6] Y. Gong, L. Liu, M. Yang, and L. D. Bourdev, "Compressing deep convolutional networks using vector quantization," *CoRR*, vol. abs/1412.6115, 2014.

[7] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," in *CVPR'2016*, 2016.

[8] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 507–516, JMLR.org, 2016.

[9] 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.

[10] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, (New York, NY, USA), pp. 1041–1049, ACM, 2017.

[11] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 5265–5274, 2018.

[12] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[13] G. Eilertsen, R. K. Mantiuk, and J. Unger, "Single-frame regularization for temporally stable cnns," *CVP*, vol. abs/1902.10424, 2019.

[14] 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.

[15] 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.

[16] 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.

[17] L. R. Elisa Maiettini, Giulia Pasquale and L. Natale, ""interactive data collection for deep learning object detectors on humanoid robots"," pp. 862–868, Nov 2017.

[18] K. Zhou, "Pytorch implementation of center loss." `https://github.com/KaiyangZhou/pytorch-center-loss`, 2018.

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