

Reliably and automatically detecting anomalies is a fundamental problem in several domains. Its applications range from financial security to medical imaging. One common way to address this problem is to frame it as a one-class classification problem: the classifier knows only examples from the normal distribution a priori, and must determine after which of the novel examples are also normal.

Despite the success of this approach in some tasks, the recent advances of Machine Learning due to Deep Neural Networks have not yet reached one-class classification techniques. Previous attempts of bringing these advances to the field required compromises, like imposing restrictions to the representational power of the neural networks. This is undesirable because one of the main strengths of the Deep Learning approach is learning useful representations from data directly, instead of relying on manual feature engineering.

We propose a method that can perform one-class classification with a different compromise. Our method imposes no restrictions in the network architecture by requiring instead labeled data from related tasks, a requirement which is not available for every scenario.

Using these related tasks, we formulate the learning of meaningful features for one-class classification as a meta-learning problem: the meta-training stage repeatedly simulates one-class classification, using the classification loss of the chosen algorithm to learn a feature representation.

We show how SVDD can be used with our method, and also propose a simpler variant based on Prototypical Networks that obtains comparable performance. This indicates that learning feature representations directly from data may be more important than which one-class algorithm we choose.

We validate our approach by adapting few-shot classification datasets to the few-shot one-class classification scenario, obtaining similar results to the state-of-the-art of traditional one-class classification, and that improves upon that of one-class classification baselines employed in the few-shot setting.

Moreover, as a practical application, we employ our method to the biometric task of on-device face verification. In this scenario, it compares unfavorably to a standard metric learning technique.

Meta-Learning for few-shot one-class classification

Gabriel Dahia Fernandes

Dissertação de Mestrado



Universidade Federal da Bahia
Instituto de Computação

Programa de Pós-Graduação em Ciência da Computação

**META LEARNING FOR FEW-SHOT
ONE-CLASS CLASSIFICATION**

Gabriel Dahia

DISSERTAÇÃO DE MESTRADO

Salvador
08 de Março de 2022

GABRIEL DAHIA

**META LEARNING FOR FEW-SHOT ONE-CLASS
CLASSIFICATION**

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Maurício Pamplona Segundo

Salvador
08 de Março de 2022

Sistema de Bibliotecas - UFBA

Dahia, Gabriel.

Meta Learning for Few-Shot One-class Classification / Gabriel Dahia – Salvador, 2022.

43p.: il.

Orientador: Prof. Dr. Maurício Pamplona Segundo.

Dissertação (Mestrado) – Universidade Federal da Bahia, Instituto de Computação, 2022.

1. Machine Learning. 2. Computer Vision. 3. Meta-Learning. I. Pamplona Segundo, Maurício. II. Universidade Federal da Bahia. Instituto de Computação. III Título.

CDD – XXX.XX

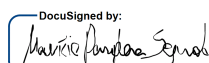
CDU – XXX.XX.XXX

“*META LEARNING FOR FEW-SHOT ONE-CLASS CLASSIFICATION*”

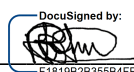
GABRIEL DAHIA FERNANDES

Dissertação apresentada ao Colegiado do Programa de Pós-Graduação em Ciência da Computação na Universidade Federal da Bahia, como requisito parcial para obtenção do Título de Mestre em Ciência da Computação.

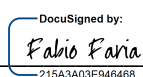
Banca Examinadora

DocuSigned by:


Prof. Dr. MAURÍCIO PAMPLONA SEGUNDO (Orientador PGCOMP)

DocuSigned by:


Prof. Dr. RUBISLEY DE PAULA LEMES (UFBA)

DocuSigned by:


Prof. Dr. FABIO AUGUSTO FARIA (UNIFESP)

ABSTRACT

Reliably and automatically detecting anomalies is a fundamental problem in several domains. Its applications range from financial security to medical imaging. One common way to address this problem is to frame it as a one-class classification problem: the classifier knows only examples from the normal distribution a priori, and must determine after which of the novel examples are also normal.

Despite the success of this approach in some tasks, the recent advances of Machine Learning due to Deep Neural Networks have not yet reached one-class classification techniques. Previous attempts of bringing these advances to the field required compromises, like imposing restrictions to the representational power of the neural networks. This is undesirable because one of the main strengths of the Deep Learning approach is learning useful representations from data directly, instead of relying on manual feature engineering.

We propose a method that can perform one-class classification with a different compromise. Our method imposes no restrictions in the network architecture by requiring instead labeled data from related tasks, a requirement which is not available for every scenario.

Using these related tasks, we formulate the learning of meaningful features for one-class classification as a meta-learning problem: the meta-training stage repeatedly simulates one-class classification, using the classification loss of the chosen algorithm to learn a feature representation.

We show how Support Vector Data Description (SVDD) can be used with our method, and also propose a simpler variant based on Prototypical Networks that obtains comparable performance. This indicates that learning feature representations directly from data may be more important than which one-class algorithm we choose.

We validate our approach by adapting few-shot classification datasets to the few-shot one-class classification scenario, obtaining similar results to the state-of-the-art of traditional one-class classification, and that improves upon that of one-class classification baselines employed in the few-shot setting.

Moreover, as a practical application, we employ our method to the biometric task of on-device face verification. In this scenario, it compares unfavorably to a standard metric learning technique.

Keywords: Machine Learning, Computer Vision, Meta-Learning

CONTENTS

Chapter 1—Introduction	1
1.1 Motivation	1
1.2 Few-shot face verification	3
1.3 Contributions	4
Chapter 2—Related work	7
2.1 One-class Classification	7
2.2 Few-shot Learning	9
2.3 Few-Shot One-Class Classification	9
2.4 Face Recognition	10
Chapter 3—Meta SVDD	13
3.1 Meta-learning One-class Classification	13
3.2 Gradient-based Optimization	15
3.3 One-class Prototypical Networks	16
Chapter 4—Face Verification as Meta One-class Classification	19
4.1 Similarities between face recognition and meta-learning	19
4.2 Face verification as few-shot one-class classification	20
4.3 VGGFace2	21
Chapter 5—Few-Shot Classification Validation	23
5.1 Evaluation Protocol	23
5.2 Datasets	23
5.2.1 Metrics and Comparison	24
5.2.2 Second experiment	24
5.3 Setup	25
5.3.1 Network Architecture	25
5.3.2 Optimization and Hyperparameters	25
5.3.3 Baselines	26
5.4 Results	26
5.4.1 First Experiment	26
5.4.2 Second Experiment	28
5.4.3 Other Experiments	29

Chapter 6—Face Verification Experiments	31
6.1 Baseline method	31
6.2 Evaluation protocol	32
6.3 Dataset	33
6.4 Setup	33
6.5 Results	34
Chapter 7—Conclusion	37
7.1 Meta-one class classification	37
7.2 Few-shot face verification	38

LIST OF FIGURES

- 1.1 Overview of the proposed method. During the meta-training stage, we emulate a training stage by first sampling X' from a distribution that is similar to the one of our target class data X . In practice we use some examples from class i - represented as the Z_i sets in the figure - from a labeled dataset Z . We then sample a minibatch of pairs (x', y') with x' being an example and y' a binary label indicating whether x' belongs to the same class as the examples in X' , sampling again from sets Z_i . Then, we use a one-class classification algorithm g (e.g. SVDD) in the features resulting from applying f_θ on the examples of X' . We use the resulting classifier to classify each example's features $f_\theta(x')$ as belonging or not to the same class as X' , and compute the binary cross entropy loss J with the true labels y' . We optimize f_θ by doing gradient descent in the value of J over many such tasks. After a final, single f_θ is learned from repeating this procedure for multiple tasks, we proceed to the training stage. There, we use f_θ to produce features for X , the data from our target class. We remark that X is disjoint from X' and that the dashed line represents the sharing of the same f_θ . Applying g to the resulting X features yields the final one-class classifier. 2
- 3.1 Figure 3.2 (a) shows in light green embeddings in 2 dimensions and in grey the center obtained by using SVDD. Notice that four points are in the dotted circle of smaller radius and four other points are outside of it, requiring the dashed circle. The smaller dashed circle near the center shows their average. Figure 3.2 (b) shows what happens with Proto Nets, where we use the average as center: the cyan points are correctly classified by taking the dotted circle as the decision boundary, and only the red circle requires a larger radius to classify. 17

- 4.1 Differences between standard face recognition and proposed few-shot face verification pipelines. The usual pipeline, shown in the left of the vertical line, usually consists of 3 steps: (1) the genuine user shares their face image with a central server, where his identity is stored, (2) a query face image is also sent to determine if the depicted person should be granted access, and (3) the server responds with its verdict, if the user is genuine, they are granted access, otherwise they are not. Notice the amount of biometric data that is exchanged with the central server - sharing is denoted by arrows crossing the dashed line. On the right of the vertical line, we show the proposed framework. (1) A trained meta one-class classifier is shared with the personal device. Notice that this is the only communication between server and device, and it flows from server to user. (2) The images of the owner of the device are used to obtain a final binary classifier, which (3) given a query image, (4) responds with its access control status. . . . 20
- 5.1 Mean Area Under the Curve (AUC) with shaded standard deviations for tasks in CIFAR datasets sorted by increasing mean value. Comparing Deep SVDD across datasets and protocols shows that the modified protocol is reasonable to evaluate few-shot one-class classification because the trend in task difficulty is similar. Within the few-shot protocol in CIFAR-FS, meta one-class classification are numerically superior, show less variance and can be meta-trained once for all tasks, with simple adaptation for unseen tasks, but require related task data. 27
- 6.1 Receiver Operating Characteristic (ROC) curve for the face verification experiment. Notice that the figure depicts only the upper-left corner of the curve – from 0.82 to 1 in the True Positive Rate (TPR) axis and from 0 to 0.2 in the False Positive Rate (FPR) axis – to make the comparison between both methods easier to analyze. The one-class prototypical network, when used for few-shot face verification, attains a lower value of TPR than the triplet-loss baseline if both operate at the same FPR for the entire displayed area of the graph. This happens even in this setting where Triplet loss does not use the support set fully, and is therefore in a disadvantage. 34

LIST OF TABLES

- 5.1 Minimum, median and maximum mean AUC alongside their standard deviation for one-class classification methods for 10 repetitions. We highlight in boldface the highest mean and others which are within one standard deviation from it. The results for the many-shot baseline Deep Convolutional Autoencoder (DCAE) in MNIST and CIFAR-10 are compiled from the table by Ruff *et al.* (RUFF *et al.*, 2018). The results for Omniglot and CIFAR-FS are for 5-shot one-class classification, and as noted before, it is unfeasible to report results for Deep SVDD in the Omniglot dataset. . . . 26
- 5.2 Mean accuracy alongside 95% confidence intervals computed over 10,000 tasks for Gaussian kernel One-class SVM with PCA, Meta SVDD and One-class Protoypical Networks. The results with highest mean and those with overlapping confidence interval with it are in boldface. We report the best result for the One-class SVM in its parameter search space, which gives it an advantage over the other two methods. Despite employing a simpler algorithm for one-class classification, One-class Prototypical networks obtain equivalent accuracy for Omniglot and better accuracy for CIFAR-FS than Meta SVDD. This indicates that learning feature representations is more important than which one-class classification algorithm we use. . . . 28
- 6.1 Equal Error Rates (EERs) for the triplet loss baseline and the one-class prototypical network methods in the face verification experiment in VGGFace2. Lower values are better, hence the baseline triplet-loss is superior to the proposed method in this experiment even in a disadvantageous setting. 35

LIST OF ACRONYMS

AUC	Area Under the Curve.....	23
CNN	Convolutional Neural Network	33
DCAE	Deep Convolutional Autoencoder.....	8
EER	Equal Error Rate	32
FPR	False Positive Rate.....	x
GAN	Generative Adversarial Network	8
GPU	Graphics Processing Unit	16
OCSVM	One-Class Support Vector Machine.....	26
PCA	Principal Component Analysis	26
ROC	Receiver Operating Characteristic.....	32
SVDD	Support Vector Data Description.....	37
SVM	Support Vector Machine.....	9
TPR	True Positive Rate	x

INTRODUCTION

1.1 MOTIVATION

One-class classification algorithms, *i.e.* classification algorithms that learn from data from a single class and must classify unseen data as either in class or not (SCHÖLKOPF et al., 2001), are the main approach to detecting anomalies from normal data. However, traditional methods scale poorly both in computational resources and sample efficiency with the data dimensions (RUFF et al., 2018).

Attempting to overcome these problems, previous work proposed using deep neural networks to learn feature representations for one-class classification. While successful in addressing some of the problems, they introduced other limitations. One problem with these methods is that some of them optimize a metric that is related, but different than their true one-class classification objective (*e.g.*, input reconstruction (SEEBÖCK et al., 2016)).

Other methods require imposing specific structure to the models, like removing biases and restricting the activation functions for the network model (RUFF et al., 2018). This is undesirable because they could diminish the representational power of the resulting networks (RUFF et al., 2018). Another approach requires using Generative Adversarial Networks (GANs) (GOODFELLOW et al., 2014; SCHLEGL et al., 2017), which are notoriously hard to optimize (ARJOVSKY; CHINTALA; BOTTOU, 2017; MESCHEDER; GEIGER; NOWOZIN, 2018).

Furthermore, like most traditional deep neural networks, these methods require thousands of samples from the target class. Unlike the success of Deep Learning techniques, however, they only obtain results that are comparable to that of the traditional baselines (RUFF et al., 2018).

Departing from previous methods, which sacrifice ease of optimization and feature representation, we propose a method whose compromise is the requirement of more data. Somewhat paradoxically we require labeled data, albeit not from the one-class classification task. Our requirement is for many pairs of binary classification tasks that are semantically similar to the task we actually care about.

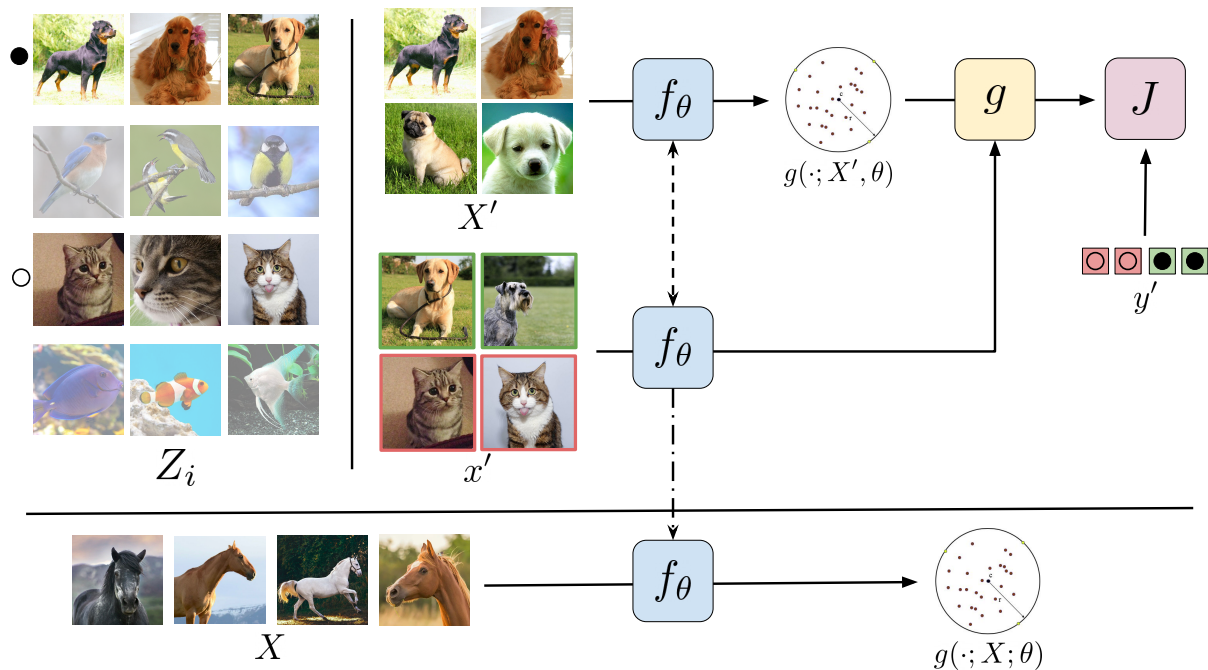


Figure 1.1 Overview of the proposed method. During the meta-training stage, we emulate a training stage by first sampling X' from a distribution that is similar to the one of our target class data X . In practice we use some examples from class i - represented as the Z_i sets in the figure - from a labeled dataset Z . We then sample a minibatch of pairs (x', y') with x' being an example and y' a binary label indicating whether x' belongs to the same class as the examples in X' , sampling again from sets Z_i . Then, we use a one-class classification algorithm g (e.g. SVDD) in the features resulting from applying f_θ on the examples of X' . We use the resulting classifier to classify each example's features $f_\theta(x')$ as belonging or not to the same class as X' , and compute the binary cross entropy loss J with the true labels y' . We optimize f_θ by doing gradient descent in the value of J over many such tasks. After a final, single f_θ is learned from repeating this procedure for multiple tasks, we proceed to the training stage. There, we use f_θ to produce features for X , the data from our target class. We remark that X is disjoint from X' and that the dashed line represents the sharing of the same f_θ . Applying g to the resulting X features yields the final one-class classifier.

A simplified overview of our method is as follows: we sample one of the binary classification tasks and define one of the classes as the normal. Then, we use a neural network to map the inputs to features, and use the features of examples in the training set to train a one-class classification model. We use this model to classify the unused examples and compute the classification loss for them using the labels. Crucially, this provides us with a signal to optimize the neural network used for feature representation. We repeat this step many times and optimize the neural network a bit for each task. The point of view is that we are optimizing over tasks from the same distribution as our main task, and thus optimizing over feature representations for one-class classification over the task

distribution.

This was already observed by the meta-learning community, where it is common to optimize over related tasks (FINN; ABBEEL; LEVINE, 2017; SNELL; SWERSKY; ZEMEL, 2017), and was the main inspiration for our method. We also enjoy other benefits from employing ideas from meta-learning: this approach improves the data efficiency of the underlying algorithm. As a result, our method obtains similar performance to traditional methods while using 1,000 times fewer data from the target class. This defines a trade-off in the availability of data from related tasks and data from the target class.

Despite the fact that our method has a very strong requirement – the availability of similar binary classification tasks with labeled examples of both classes – there are some natural domains where this is satisfied. For example, in fraud detection, we could use normal activity from other users and create related tasks that consist of identifying if the activity came from the user or not, while still employing and optimizing one-class classification.

We describe an instance of our method, the *Meta Support Vector Data Description*, obtained by using the SVDD (TAX; DUIN, 2004) as the one-class classification algorithm. We chose the SVDD based on its simplicity, intuitiveness and the success it obtains in low-dimension input spaces. It is also very similar to the One-Class Support Vector Machine (OCSVM), and enjoys similar theoretical guarantees (TAX; DUIN, 2004).

The main drawback of the SVDD is that it requires solving a quadratic program to compute the center of its hypersphere. We noted that if we instead used the centroid of the inputs, we would obtain a sphere with greater radius but it would be much simpler to compute. Not only that, but one of the most successful methods for meta-learning is based on the same idea (SNELL; SWERSKY; ZEMEL, 2017). Despite its simplicity, this method obtains comparable performance to Meta SVDD.

1.2 FEW-SHOT FACE VERIFICATION

Inspired by the success of Deep Learning techniques for computer vision (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) in turning automated face recognition from a research problem into a viable application (SCHROFF; KALENICHENKO; PHILBIN, 2015; TAIGMAN et al., 2014), we considered a practical application for our one-class classification methods: addressing some of the privacy concerns raised by commercial application of facial recognition technology.

These concerns started when civil society organizations noted that the wide availability of faces in the internet, driven by mass use of social networks, and demanded regulatory action. This led San Francisco and then California to ban the use of face recognition technology by law-enforcement agencies (CAGLE, 2019).

Similar concerns and others regarding fairness towards subrepresented peoples (RAJI et al., 2020) have also led to the removal of the massive online datasets previously used for face recognition (MURGIA, 2019a). Not only that, but now there are also tools that allow searching for faces in most of the reachable internet (PIMEYES, 2017).

These should not impact the usage of face recognition technology for personal use, which does not require sharing any biometric information with other parties. Nonetheless,

face recognition software is now developed using metadata from social networks and mostly unknowing users. The collection of this data into servers belonging to companies that do not value or respect user privacy resulted in the reduction of research in face recognition for personal use.

In this work, we consider the usage of face recognition for single-user authentication in a privacy-preserving manner. In this setting, it is usual to assume the verification system has many photos of the registered user’s face, *e.g.* the owner of the device has a personal photo library with more than one picture with their faces on it. On the other hand, people querying the system provide only a single image to request access

Therefore, we propose to combine Deep Convolutional Neural Network (CNN) with our proposed one-class classification methods and compare them with a traditional baseline for face recognition in this understudied setting. Crucially, the few-shot nature of our work means adaptation should have low computational cost, should not require specific hardware, and means it must rely on few data for the unseen task.

We therefore propose a protocol for evaluation of few-shot face verification, and implement it using the VGGFace2 dataset (CAO et al., 2018), known for its balance between large scale and comparatively low noise level. We use the resulting benchmark to compare three methods: our two, Meta Support Vector Data Description (Meta SVDD) and One-class Prototypical Networks, and an adaptation of a traditional method for face recognition, deep metric learning using Triplet-Loss (SCHROFF; KALENICHENKO; PHILBIN, 2015) to the few-shot face verification setting.

1.3 CONTRIBUTIONS

- We show how to learn a feature representation for one-class classification (chapter 3) by defining an estimator for the classification loss of such algorithms (section 3.1). We also describe how to efficiently backpropagate through the objective when the chosen algorithm is the SVDD method, so we can parametrize the feature representation with deep neural networks (section 3.2). The efficiency requirement to train our model serves to make it work when there are as little as 5 examples, which puts it in the few-shot regime.
- We simplify Meta SVDD by replacing how the center of its hypersphere is computed. Instead of solving a quadratic optimization problem to find the weight of each example in the center’s averaging, we remove the weighting and make the center the result of an unweighted average. In other words, we use the prototype or centroid of the examples’ features (Section 3.3). The resulting method, called One-class Prototypical Networks, are simpler, have lower computational complexity and more stable training dynamics than Meta SVDD.
- We also show that our method has promising empirical performance by adapting two few-shot classification datasets to the one-class classification setting and obtaining comparable results with the state-of-the-art of the many-shot setting (chapter 5). Our results indicate that learning the feature representations may compensate for the simplicity of replacing SVDD with feature averaging and that our approach is a

viable way to replace data from the target class with labeled data from related tasks. Code to reproduce our experiments and methods is also made publicly available¹.

- The last contribution is testing whether our method is applicable to a practical, useful task: few-shot face verification. We first detail how we frame this problem as a meta one-class classification problem, and how the few-shot nature of our method is useful for it (chapter 4). We then conduct experiments in a large face recognition dataset, adapted to our task, comparing a traditional and well-tested baseline method to the stronger of our one-class classification methods (chapter 6). The baseline obtains a better result, which we interpret as indication that more could be learned from metric learning in the meta-learning community than the other way around.

This dissertation is organized as follows: chapter 2 reviews related work; chapter 3 briefly reviews the SVDD method and shows how we can use it for meta-learning, thus obtaining Meta SVDD; Section 3.3 shows that simplifying the method, we obtain a variant of the Prototypical Networks method for one-class classification; chapter 5 details our experiments and the observed results; chapter 6 explains our protocol for testing meta one-class classification in the few-shot face verification setting; and chapter 7 discusses the method and future work.

¹https://github.com/gdahia/meta_occ

RELATED WORK

2.1 ONE-CLASS CLASSIFICATION

We briefly survey the main ideas related to our work in the one-class classification literature. A more detailed treatment of this subject can be found in the recent survey of Perera *et. al* (PERERA; OZA; PATEL, 2021).

The Support Vector Data Description (SVDD) (TAX; DUIN, 2004), reviewed in chapter 3, is closely related to the One-Class Support Vector Machine (OCSVM) (SCHÖLKOPF et al., 2001). Whereas the SVDD finds a hypersphere to enclose the input data, the OCSVM finds a maximum margin hyperplane that separates the inputs from the origin of the coordinate system. Like the SVDD, it can also be formulated as a quadratic program, solved in kernelized form, and use slack variables to account for outliers in the input data. In fact, when the chosen kernel is the commonly used Gaussian kernel, both methods are equivalent (SCHÖLKOPF et al., 2001).

Besides their equivalence in that case, the OCSVM more generally suffers from the same limitations as the SVDD. Both require explicit feature engineering (*i.e.* it prescribes no way to formulate f_θ), and it scales poorly both with the number of samples and the dimension of the data.

The limitations of SVDD and OCSVMs led to the development of *deep* approaches to one-class classification, where the previous approaches are known as *shallow* because they do not rely on deep (*i.e.* multi-layered) neural networks for feature representation.

Most previous approaches that use deep neural networks to represent the input feature for downstream use in one-class classification algorithms are trained with a surrogate objective, like the representation learned for input reconstruction with deep autoencoders (HINTON; SALAKHUTDINOV, 2006).

Autoencoder methods learn feature representations by requiring the network to reconstruct inputs while preventing it to learn the identity function. These are usually divided into an encoder, tasked with converting an input example into an intermediate representation, and a decoder, that gets the representation and must reconstruct the input (GOODFELLOW; BENGIO; COURVILLE, 2016).

The idea is that if the identity function cannot be learned, then the representation has captured semantic information of the input that is sufficient for its partial reconstruction and other tasks. How the identity function is prevented determines the type of autoencoder and many options exist: by reducing the dimensions of or imposing specific distributions to the intermediate representations, by adding a regularization term to the model’s objective, or by corrupting the input with noise (GOODFELLOW; BENGIO; COURVILLE, 2016).

Philipp Seeböck *et al.* (SEEBÖCK *et al.*, 2016) train a Deep Convolutional Autoencoder (DCAE) in images for the target class, here healthy retinal image data, and after that the decoder is ignored and an OCSVM is trained on the resulting intermediate representations. The main issue with this approach is that the objective of autoencoder training does not assure that the learned representations are useful for classification.

A related approach is to reuse features from networks trained for multiclass classification. Oza and Patel (OZA; PATEL, 2019) remove the softmax layer of a Convolutional Neural Network (CNN) (LECUN *et al.*, 1990) trained in the ImageNet dataset (DENG *et al.*, 2009) as its feature extractor. The authors then train the fully-connected layers of the pre-trained network alongside a new fully connected layer tasked with discriminating between features from the target class and data sampled from a spherical Gaussian distribution; the convolutional layers are not updated.

AnoGANs (SCHLEGL *et al.*, 2017) are trained like Generative Adversarial Networks (GANs) (GOODFELLOW *et al.*, 2014) to generate samples from the target class. After that, gradient descent is used to find the sample in the noise distribution that best reconstructs the unseen example to be classified, which is equivalent to approximately inverting the generator using optimization. The classification score is the input reconstruction error, which assumes pixel-level similarity determines membership in the target class.

Like the method we propose, Deep SVDD (RUFF *et al.*, 2018) attempts to learn feature representations for one-class classification from the data using gradient-based optimization with a neural network model. It consists of directly reducing the volume of a hypersphere containing the features, and in that it is a deep version of the original SVDD.

Deep SVDD’s algorithm relies on setting the centers every few iterations with the mean of the features from a forward pass instead of computing the minimum bounding sphere. Since their objective is to minimize the volume of the hypersphere containing the features, the algorithm must avoid the pathological solution of outputting a constant function. This requires imposing architectural constraints on the network, the stronger of which is that the network’s layers can have no bias terms. The authors also initialize the weights with those of an encoder from a trained autoencoder.

One advantage of Deep SVDD over our work is that it does not require data from tasks from a similar distribution: it is trained only on the target class data. While this is an advantage, there is a downside to it. It is not clear for us, reading the paper describing Deep SVDD, how to know for how long to train a Deep SVDD model, how to tune its many hyperparameters, or what performance to expect of the method in unseen data. These are usually done with computing useful metrics in a validation set. However, for

Deep SVDD, the optimal value can be reached for pathological solutions, so a validation set is not useful.

Ruff *et al.* (RUFF et al., 2018) prove that using certain activation functions or keeping bias terms allow the model to learn the constant function but they do not prove the reciprocal, *i.e.* they do not prove that constant functions cannot be learned by the restricted models. The authors also do not analyze which functions are no longer learnable when the model is restricted as such.

2.2 FEW-SHOT LEARNING

The main inspiration for the ideas in our paper besides Deep SVDD came from the field of meta-learning, in particular, that of few-shot classification. Roughly speaking, few-shot classification means there are as few as 5 examples to learn a given task. The scarcity of data is compensated by the abundance of semantically similar tasks for meta-training.

Meta-training, on its turn, consists on either learning a good initialization so that the few examples suffice to train a classifier – usually called optimization-based learning and best exemplified by Model Agnostic Meta-Learning (FINN; ABBEEL; LEVINE, 2017) – or learning a metric space that transfers to the unseen task – the metric learning Meta-Learning approach started by Prototypical Networks (SNELL; SWERSKY; ZEMEL, 2017). Our approach is an example of the latter, and we focus our review on such methods.

Prototypical Networks (SNELL; SWERSKY; ZEMEL, 2017) are few-shot classifiers that create prototypes from few labeled examples and use their squared Euclidean distances to an unseen example as the logits to classify it as one of their classes. We first saw the idea of learning the feature representation from similarly distributed tasks and of using the squared distances in this paper. They also propose feature averaging as a way to summarize class examples and show its competitive performance despite its simplicity; One-class Prototypical Networks are the one-class variant of this method.

Recently, Lee *et al.* (LEE et al., 2019) proposed to learn feature representations for few-shot classification convex learners, including multi-class Support Vector Machines (SVMs) (CORTES; VAPNIK, 1995), with gradient-based optimization. Their work is similar to ours in its formulation of learners as quadratic programs, and in solving these with quadratic programming layers but it does not address one-class classification.

2.3 FEW-SHOT ONE-CLASS CLASSIFICATION

Concurrent with our work, others have started investigating the use of few-shot classification methods for the one-class classification task. Most similar to our work, Oladosu *et al.* (OLADOSU et al., 2020) propose to perform meta-learning in a setting that is similar to ours. Their method is trained with multiple supervised tasks so it can perform on unseen tasks with only positive example at deploy time.

The main differences between our methods is that theirs uses set equivariant networks to learn representations from the positive examples, whereas we use either SVDD or a Prototypical Network layer. This means they require classifying the entire positive

dataset for each query point, but doing so allows them to avoid defining an explicit metric in feature space.

Like us, they also evaluate their method by adapting datasets with supervision to the meta-learning one-class setting by splitting them over the labels instead of over examples.

2.4 FACE RECOGNITION

CNNs improved computer vision drastically (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), and it did not take long for the automatic face recognition community to take notice. Previously, successful face recognition methods required acquisition of face images in environments with controlled pose, lighting and facial expressions. Deep learning techniques allowed these constraints to be lifted, as noted by the increase in performance in the then default benchmark for face recognition in-the-wild with the introduction of these techniques (TAIGMAN et al., 2014; SCHROFF; KALENICHENKO; PHILBIN, 2015). This has broadened the scope for which reliable biometrics can be used in security; for example, we no longer require contact-based modalities like fingerprints to reliably recognize individuals. These advances have recently been surveyed in much more detail (WANG; DENG, 2021).

Most successful work formulates the problem of face recognition as a variation of a standard pipeline consisting of: image acquisition, face normalization, feature representation, and matching. The achievement of deep learning in this area is the ability to reduce the feature representation step to a stochastic optimization problem relying on large amounts of data. Notably, the field of metric learning allows one to solve feature representation and example matching at once, by learning features that map semantic distance onto a simple space, like Euclidean space. Several works approach the problem like these, proposing variations on the specifics of embedding the feature space into the simple space or heuristics to improve separability therein (TAIGMAN et al., 2014; SCHROFF; KALENICHENKO; PHILBIN, 2015; WEN et al., 2016; DENG et al., 2019; WANG et al., 2018).

The Center-Loss method (WEN et al., 2016), in particular, is very similar to the One-Class Prototypical networks we investigate in this work. There are some subtle differences. The first is that Center-Loss keeps one center for each identity during the entire training, and updates them using a weighted average, which requires another hyperparameter; one-class prototypical networks compute centers or prototypes from mini-batch examples. The other difference is that there is no separation between query and support examples when training with Center-Loss, which was previously thought as problematic for few-shot classification (SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017) but has since been reevaluated (LAENEN; BERTINETTO, 2021).

However, most of this progress has been seriously challenged by imposing fairer comparisons and a more rigorous benchmark for metric learning in computer vision (MUSGRAVE; BELONGIE; LIM, 2020b). More traditional methods, like contrastive loss (TAIGMAN et al., 2014) and triplet-loss (SCHROFF; KALENICHENKO; PHILBIN, 2015) have been shown to be as competitive or more than proposed more recent methods. We therefore focus our comparison with triplet-loss, a widely known and effective baseline for

face recognition (SCHROFF; KALENICHENKO; PHILBIN, 2015; MUSGRAVE; BELONGIE; LIM, 2020b).

Interestingly, most face verification benchmarks evaluate scenarios that do not match the specific setting of few-shot face verification. When human performance was surpassed in one-to-one face verification in-the-wild (HUANG et al., 2007; TAIGMAN et al., 2014), the newer benchmarks started focusing on more challenging tasks. Nowadays, the focus is improvement on the Youtube Faces Dataset (Wolf; Hassner; Maoz, 2011), and the IARPA Janus Benchmark (KLARE et al., 2015), which evaluate few-shot face verification; on the MegaFace dataset (KEMELMACHER-SHLIZERMAN et al., 2016), which evaluates open-set face identification. The most relevant benchmark is the MSCelebFaces dataset with its few-shot challenge (GUO; ZHANG, 2017). It evaluates closed-set few-shot identification, a setting which differs from what we study in two aspects: (1) it proposes to match multiple genuine users providing few samples of each, instead of one, and (2) it ensures that all query users will be in the known dataset, whereas we use only unknown users for testing.

META SVDD

The Support Vector Data Description (SVDD) method (TAX; DUIN, 2004) computes the hypersphere of minimum volume that contains every point in the training set. The idea is that only points inside the hypersphere belong to the target class, so we minimize the sphere’s volume to reduce the chance of including points that do not belong in the target class.

Formally, the radius $R(X, c; \theta)$ of the hypersphere centered at $c \in \mathbb{R}^d$ covering the training set X transformed by $f_\theta : \mathbb{R}^D \rightarrow \mathbb{R}^d$ is

$$R(X, c; \theta) = \max_{x \in X} \|f_\theta(x) - c\|. \quad (3.1)$$

The SVDD objective is to find the center c^* that minimizes the radius of such a hypersphere, *i.e.*

$$c^* = \arg \min_c R(X, c; \theta). \quad (3.2)$$

Finally, the algorithm determines that a point x' belongs to the target class if

$$\|f_\theta(x') - c^*\| \leq R(X, c^*; \theta). \quad (3.3)$$

The SVDD objective, however, does not specify how to optimize the feature representation f_θ . Previous approaches include using dimensionality reduction with Principal Component Analysis (PCA) (RUFF et al., 2018), using a Gaussian kernel with the kernel trick (TAX; DUIN, 2004), or using features learned with unsupervised learning methods, like deep belief networks (ERFANI et al., 2016). We take a different approach: Our goal is to learn f_θ for the task, and we detail how next.

3.1 META-LEARNING ONE-CLASS CLASSIFICATION

Our objective is to learn a f_θ such that the minimum volume hypersphere computed by the SVDD covers only the samples from the target class. We, therefore, divide the learning problem into two stages. In the *meta-training* stage, we learn the feature representation

f_θ . Once we learn f_θ , we use it to learn a one-class classifier using the chosen algorithm (in this case, SVDD) from the data of the target class in the *training* stage. This is illustrated in Figure 1.1.

Notice how both the decision on unseen inputs (Equation 3.3) and the hypersphere’s center c^* (Equation 3.2) depend on f_θ . Perfectly learning f_θ in the meta-training stage would map *any* input distribution into a space that can be correctly classified by SVDD, and would therefore not depend on the given data X nor on what is the target class; that would be learned by the SVDD after transforming X with f_θ in the subsequent training stage. We do not know how to learn f_θ perfectly but the above observation illustrates that we do not need to learn it with data from the target class.

With that observation, we can use the framework of nested learning loops (RAGHU et al., 2019) to describe how we propose to learn f_θ :

- *Inner loop*: Use f_θ to transform the inputs, and use SVDD to learn a one-class classification boundary for the resulting features.
- *Outer loop*: Learn f_θ from the classification loss obtained with the SVDD.

We use the *expected* classification loss in the outer loop. With this, we can use data that comes from the same distribution as the data for the target class, but with different classification tasks. To make this definition formal, first, let g be a one-class classification function parametrized by θ which receives as inputs a subset of examples from the target class X' and an example x' , and outputs the probability that x' belongs to the target class. For a suitable classification loss J , our learning loss is

$$\mathcal{L}(\theta) = \mathbb{E}_{X' \sim \mathcal{D}_X} [\mathbb{E}_{(x', y') \sim \mathcal{D}_{Z|X'}} [J(g(x', X'; \theta), y')]] \quad (3.4)$$

where y' is a binary label indicating whether x' belongs to the same distribution of X' or not. The outer expectation of Equation 3.4 defines a one-class classification task, and the inner expectation is over labeled examples for this task (hence the dependency on X' for the labeled example distribution $\mathcal{D}_{Z|X'}$). Since we do not have access to the distribution \mathcal{D}_X nor we have access to $\mathcal{D}_{Z|X}$, we approximate it with related tasks. Intuitively, the closer the distribution of the tasks we use to approximate it, the better our feature representation.

To compute this approximation in practice, we require access to a labeled multiclass classification dataset $Z = \{(x_1, y_1), \dots, (x_N, y_N)\}$, where $x_i \in \mathbb{R}^D$ is the i^{th} element and $y_i \in Z$ its label, that has a distribution similar to our dataset X , but is disjoint from it (*i.e.* none of the elements in X are in Z and none of its elements belong to any of the classes in Z). Datasets like Z are common in the meta-learning or few-shot learning literature, and their existence is a standard assumption in previous work (SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017; LEE et al., 2019). However, this restricts the tasks to which our method can be applied to those that have such related data available.

We then create the datasets Z_1, \dots, Z_k from Z by separating its elements by class, *i.e.*

$$Z_i = \{x_j \mid (x_j, i) \in Z\}. \quad (3.5)$$

We create the required binary classification tasks by picking Z_i as the data for the target class, and the examples from Z_j , $j \neq i$, to be the input data from the negative class. Finally, we approximate the expectations in Equation 3.4 by first sampling mini-batches of these binary classification tasks and then averaging over mini-batches of labeled examples Z' from each of the sampled tasks. By making each sampled X' have few examples (*e.g.* 5 or 20), we not only make our method scalable but we also learn f_θ for *few-shot one-class classification*.

In the next section, we define a model for f_θ and show how to optimize it over Equation 3.4.

3.2 GRADIENT-BASED OPTIMIZATION

If we choose f_θ to be a neural network, it is possible to optimize it to minimize the loss in Equation 3.4 with gradient descent as long as J and g are differentiable and have meaningful gradients because of the chain rule of calculus. J can be the standard binary cross-entropy between the data and model distributions (GOODFELLOW; BENGIO; COURVILLE, 2016).

We also modify the SVDD to satisfy the requirements of the g function. Neither how it computes the hypersphere’s center, by solving an optimization problem (Equation 3.2), nor its hard, binary decisions (Equation 3.3) are immediately suitable for gradient-based optimization.

To solve the hard, binary decisions problem, we adopt the approach of Prototypical Networks (SNELL; SWERSKY; ZEMEL, 2017) and consider the squared distance from the features $f_\theta(x')$ to the center c^* (the left-hand side of Equation 3.3) as the input logits for a logistic regression model. Doing this not only solves the problem of uninformative gradients coming from the binary outcomes of SVDD but also simplifies its implementation in modern automatic differentiation/machine learning software, *e.g.* PyTorch (PASZKE et al., 2019). As our logits are non-negative, using the sigmoid function σ to convert logits into probabilities would result in probabilities of at least 0.5 for every input, so we replace it with the tanh and keep the binary cross-entropy objective otherwise unchanged.

As for how to compute c^* in a differentiable manner, we can write it as the weighted average of the input features

$$c^* = \sum_{i=1}^n \alpha_i f_\theta(x_i) \quad (3.6)$$

where the weights α are the solution of the following quadratic programming problem, which is the dual of the problem defined in Equation 3.2 (ELZINGA; HEARN, 1972; TAX; DUIN, 2004)

$$\max_{\alpha} \alpha^T \text{diag}(K) - \alpha^T K \alpha \quad (3.7)$$

$$\text{subject to } \sum_{i=1}^n \alpha_i = 1 \quad (3.8)$$

$$0 \leq \alpha_i, i \in \{1, \dots, n\} \quad (3.9)$$

and

$$K_{i,j} = f_{\theta}(x_i)^T f_{\theta}(x_j) \quad (3.10)$$

is the kernel matrix of f_{θ} for input set X . Hence, we can interpret f_{θ} as a data-dependent kernel function trained on a related distribution.

Despite such quadratic programs not having known analytical solutions and requiring a projection operator to unroll its optimization procedure because of its inequality constraints, the quadratic programming layer (AMOS; KOLTER, 2017) can efficiently backpropagate through its solution and supports Graphics Processing Unit (GPU) usage.

Still, the quadratic programming layer has complexity $O(m^3)$ for m optimization variables (AMOS; KOLTER, 2017); in the case of Meta SVDD, m is equal to the number of examples in X during training (LEE et al., 2019). As the size of the network is constant, this is an upperbound on the overall complexity of performing a training step in the model. Since we keep the number of examples small, 5 to 20, the runtime is dominated by the computation of f_{θ} .

In practice, we follow previous work that uses quadratic programming layers (LEE et al., 2019) and we add a small stabilization value $\lambda = 10^{-6}$ to the diagonals of the kernel matrix (Equation 3.10), *i.e.*

$$K' = K + \lambda I \quad (3.11)$$

and we use K' in Equation 3.7. Not adding this stabilization term results in failure to converge in some cases.

Using the program defined by objective 3.7, and constraints 3.8 and 3.9 to solve SVDD also allows us to use the kernel trick to make K non-linear with regards to f_{θ} (TAX; DUIN, 2004). We believe this would not add much since using a deep neural network to represent f_{θ} can handle the non-linearities that map the input to the output, in theory.

SVDD (TAX; DUIN, 2004) also introduce slack variables to account for outliers in the input set X . Since our setting is few-shot one-class classification, we do not believe these would benefit the method’s performance because we think outliers are unlikely in such small samples. We leave the analysis to confirm or refute these conjectures to future work.

3.3 ONE-CLASS PROTOTYPICAL NETWORKS

The only reason to solve the quadratic programming problem defined by objective 3.7 and constraints 3.8 and 3.9 is to obtain the weights α for the features of each example in Equation 3.6.

We experiment with replacing the weights α in Equation 3.6 by uniform weights $\alpha_i = 1/n$. The center c^* then becomes a simple average of the input features

$$c^* = \frac{1}{n} \sum_{i=1}^n f_{\theta}(x_i) \quad (3.12)$$

and we no longer require solving the quadratic program. The remainder of the method, *i.e.* its training objective, how tasks are sampled, etc, remains the same. This avoids the

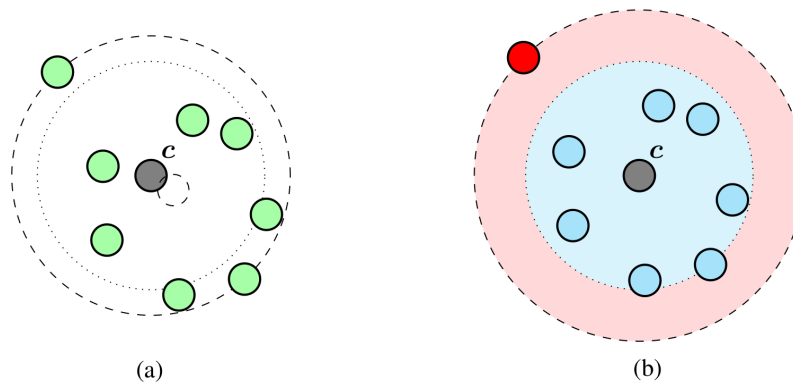


Figure 3.1 Figure 3.2 (a) shows in light green embeddings in 2 dimensions and in grey the center obtained by using SVDD. Notice that four points are in the dotted circle of smaller radius and four other points are outside of it, requiring the dashed circle. The smaller dashed circle near the center shows their average. Figure 3.2 (b) shows what happens with Proto Nets, where we use the average as center: the cyan points are correctly classified by taking the dotted circle as the decision boundary, and only the red circle requires a larger radius to classify.

cubic complexity in the forward pass, and the destabilization issue altogether. We call this method One-class Prototypical Networks because the method can be cast as learning binary Prototypical Networks (SNELL; SWERSKY; ZEMEL, 2017) with a binary cross-entropy objective. Figure 3.1 shows an illustration of the difference between the two methods in two dimensions.

Despite being a simpler method than Meta SVDD, we conjecture that learning f_θ to be a good representation for One-class Prototypical Networks can compensate its algorithmic simplicity so that performance does not degrade.

FACE VERIFICATION AS META ONE-CLASS CLASSIFICATION

4.1 SIMILARITIES BETWEEN FACE RECOGNITION AND META-LEARNING

Beyond both the face recognition and meta-learning community having discovered each other's techniques independently, like in the contrastive loss (TAIGMAN et al., 2014; LAKE; SALAKHUTDINOV; TENENBAUM, 2015) and the similarities between center loss (WEN et al., 2016) and prototypical networks (SNELL; SWERSKY; ZEMEL, 2017), there is a natural connection between both areas.

Whereas in traditional computer vision applications we are given a fixed set of categories and for a given unseen example must determine to which class it belongs (DENG et al., 2009; KRIZHEVSKY; SUTSKEVER; HINTON, 2012), in modern face recognition we are given a dataset of labeled faces from which to learn the abstract task of face recognition, and then a test set of face images of individuals not in the training set and must determine which of these examples belong to the same person (HUANG et al., 2007; GUO et al., 2016; CAO et al., 2018).

In meta-learning, we also assume a disjoint set of training and test tasks. We must then use the labeled training tasks to learn a task representation, and use this to generalize to unseen test tasks (LAKE; SALAKHUTDINOV; TENENBAUM, 2015; SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017). Oftentimes, generalization is only attempted in the same domain, *i.e.* from computer vision tasks to other computer vision tasks, from natural language processing tasks to other such tasks.

We propose, then, to consider each face identity as one learning task. More specifically, the task is determined by a given identity I with enough of examples of their face, and consists of labeling unseen examples as belonging to I or not. Crucially, we also rely on the ability of a learned classifier to generalize across unseen identities/tasks; otherwise, we would require knowing all identities beforehand during the training phase, which is incompatible with the current privacy requirements dictated by society.

4.2 FACE VERIFICATION AS FEW-SHOT ONE-CLASS CLASSIFICATION

The above formulation has no implications whatsoever in a different protocol for face recognition, it just highlights the similarities between these two areas. Most commercial applications attempt to solve the entire problem of face recognition at once from a large enough training dataset, and then use collected image(s) from the user to specialize the classifier to them. Also given the recent privacy constraints, ceding one’s images and associating them to access control is undesirable. Our proposed solution is then to allow the same classifier to be specialized to each user by the user herself. Doing that, no images are shared with the classifier provider.

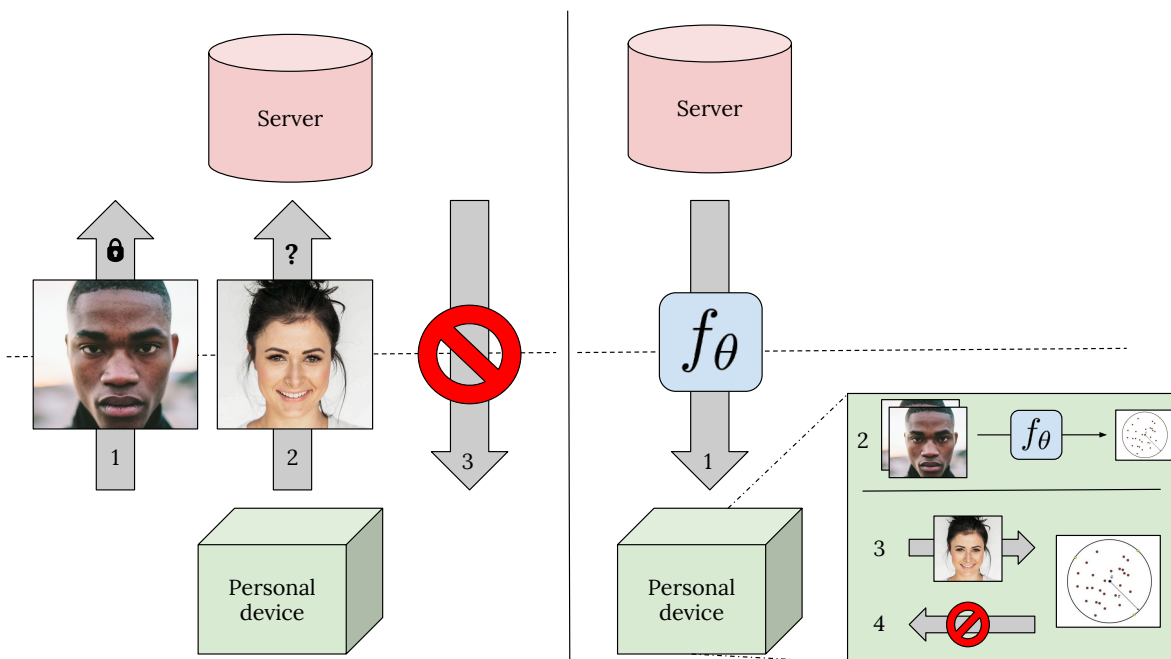


Figure 4.1 Differences between standard face recognition and proposed few-shot face verification pipelines. The usual pipeline, shown in the left of the vertical line, usually consists of 3 steps: (1) the genuine user shares their face image with a central server, where his identity is stored, (2) a query face image is also sent to determine if the depicted person should be granted access, and (3) the server responds with its verdict, if the user is genuine, they are granted access, otherwise they are not. Notice the amount of biometric data that is exchanged with the central server - sharing is denoted by arrows crossing the dashed line. On the right of the vertical line, we show the proposed framework. (1) A trained meta one-class classifier is shared with the personal device. Notice that this is the only communication between server and device, and it flows from server to user. (2) The images of the owner of the device are used to obtain a final binary classifier, which (3) given a query image, (4) responds with its access control status.

The proposed method for meta one-class classification also has other benefits from the user’s point of view. By making the method few-shot, the user need only to use few of its face images; this also increases the runtime of classifier specialization. Figure 4.1 shows the differences between the proposed pipeline and the usual one.

We formally frame face verification as an instance of our method as follows:

1. We have a dataset of labeled face identities; these serve as the related tasks Z .
2. The target user is identified with the dataset X .
3. The remainder of the method remains unchanged.

To simulate this scenario and validate our approach, we adapt a traditional face recognition dataset to this setting, the VGGFace2 dataset (CAO et al., 2018).

4.3 VGGFACE2

The VGGFace2 dataset is an image dataset for face recognition, containing 3.31 million face images with varying age, ethnicity, illumination, and pose of 9131 subjects. It provides not only a large number of different people, but several examples per person, with an average of 326.6 for each subject (CAO et al., 2018).

The maintainers of the dataset propose a default split for it, with 1000 identities reserved for testing and the remainder for training. Importantly, this is a subject disjoint split, *i.e.* there is no identity in both training and testing subsets.

Despite not being the dataset with neither the greatest number of individuals, nor the one with the most images absolutely or in average, the VGGFace2 dataset balances these two attributes well, with enough images to train data-hungry CNN-based methods. Furthermore, the alternative dataset with more images, MSCeleb1M (GUO et al., 2016) is no longer publicly available (MURGIA, 2019b).

An even better reason to use VGGFace2 in our setting is that it is the largest dataset with moderate amount of noise in its identity labels (CAO et al., 2018). Since we believe the total number of examples and the number of examples per individual is enough to empirically validate our method, and its labels are less noisier than other datasets of similar size, we believe VGGFace2 is the best dataset in which to run our experiments.

FEW-SHOT CLASSIFICATION VALIDATION

5.1 EVALUATION PROTOCOL

Our first experiment is an adaptation of the evaluation protocol of Deep Support Vector Data Description (SVDD) (RUFF et al., 2018) to the few-shot setting to compare Meta SVDD with previous work. This experiment is a comparison between deep one-class classification methods, and we adapt it to also assess the potential of Deep SVDD in the few-shot setting. The reason for not using a few-shot baseline is because while we developed our methods, we were not aware of methods operating simultaneously in the few-shot and one-class classification settings.

The original evaluation protocol proposed by the authors of Deep SVDD consists of picking one of the classes of the dataset, training the method in the examples in the training set (using the train-test split proposed by the maintainers), and using all the examples in the test set to compute the mean and standard deviation of the Area Under the Curve (AUC) of the trained classifier. The mean is computed over 10 repetitions in the MNIST (LeCun et al., 1998) and CIFAR-10 (KRIZHEVSKY, 2009) datasets, and they were chosen by the authors of Deep SVDD (RUFF et al., 2018).

We modified the protocol because there are only 10 classes in these datasets, which is not enough for meta-learning one-class classifiers. These methods usually require many more tasks to learn feature representations that transfer from one task to the other (SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017; LEE et al., 2019); in our case, each task corresponds to one class, and so 10 classes do not suffice. This illustrates the trade-off introduced by our approach: Despite requiring many fewer examples per class, it requires many more classes. Our modifications are only to address the number of classes and we tried to keep the protocol as similar as possible to make the results more comparable.

5.2 DATASETS

The first modification is the replacement of CIFAR-10 dataset by the CIFAR-FS dataset (BERTINETTO et al., 2019), a new split of CIFAR-100 for few-shot classification in

which there is no class overlap between the training, validation and test sets. CIFAR-FS has 64 classes for training, 16 for validating, and 20 for testing, and each class has 600 images. This results in a split of 64% of the data for training, 16% for validation and 20% for testing in the CIFAR-FS dataset.

No such split is possible for MNIST because there is no fine-grained classification like in the case of the CIFAR-10 and CIFAR-100 datasets. Therefore, we use the Omniglot dataset (LAKE; SALAKHUTDINOV; TENENBAUM, 2015), which is considered the “transposed” version of the MNIST dataset because it has many classes with few examples instead of the many examples in the 10 classes of MNIST. This dataset consists of 20 images of each of its 1623 handwritten characters, which are usually augmented with four multiples of 90° to obtain $1623 \times 4 = 6492$ classes (VINYALS et al., 2016; BERTINETTO et al., 2019; SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017). We follow the pre-processing and dataset split proposed by VINYALS et al. (2016), *i.e.* we resize the images to 28×28 pixels, and use 4800 classes for training and 1692 for testing; this is nowadays standard in few-shot classification work (FINN; ABBEEL; LEVINE, 2017; SNELL; SWERSKY; ZEMEL, 2017; BERTINETTO et al., 2019). Hence, we have approximately 73% of the data for training and 26% of the data for testing in the Omniglot dataset.

Note that these many examples are only used to sample tasks with few examples each. During evaluation, we only use 5 examples unless otherwise noted.

We also modify the number of elements per class in the test set evaluation. Since there are many classes and we are dealing with few-shot classification, we use only two times the number of examples in X for the target and for the negative class, *e.g.* if the task is 5-shot learning, then there are 10 examples from the target class and 10 examples from the negative class for evaluation.

5.2.1 Metrics and Comparison

Another modification is that since there are only 10 classes in MNIST and CIFAR-10, Deep SVDD (RUFF et al., 2018) reports the AUC metrics for each class. This is feasible for CIFAR-FS, which has 20 testing classes, but not for Omniglot, which has 1692. We summarize these statistics by presenting the minimum, median, and maximum mean AUC alongside their standard deviations.

To better compare the previous methods with ours in the few-shot setting, we evaluate the state-of-the-art method for general deep one-class classification, Deep SVDD (RUFF et al., 2018), in our modified protocol. We run the evaluation protocol in CIFAR-FS using only 5 images for training, and we evaluate it using 10 images from the target class and 10 images from a negative class. We do this 10 times for each pair of the 20 test classes to compute mean and standard deviation statistics for the AUC. We do not do this for Omniglot because it would require training more than 1692 Deep SVDD models.

5.2.2 Second experiment

We also conduct a second experiment, based on the standard few-shot classification experiment (SNELL; SWERSKY; ZEMEL, 2017; FINN; ABBEEL; LEVINE, 2017): we

evaluate the mean 5-shot one-class classification accuracy over 10,000 episodes of tasks consisting of 10 examples from the target class and 10 examples from the negative class. We use this experiment to compare with a shallow baseline, PCA and Gaussian kernel One-class SVM (SCHÖLKOPF et al., 2001), and One-class Prototypical Network. We use the increased number of episodes to compute 95% confidence intervals like previous work for few-shot multiclass classification (BERTINETTO et al., 2019; LEE et al., 2019).

5.3 SETUP

5.3.1 Network Architecture

We parametrize f_θ with the neural network architecture model introduced by VINYALS et al. (2016). This is standard practice in the meta-learning literature, as this architecture is commonly used in other few-shot learning work (FINN; ABBEEL; LEVINE, 2017; SNELL; SWERSKY; ZEMEL, 2017). This Convolutional Neural Network (CNN) has four convolutional blocks with number of filters equal to 64, and each block is composed of a 3×3 kernel, stride 1, “same” 2D convolution, batch normalization (IOFFE; SZEGEDY, 2015), followed by 2×2 max-pooling and ReLU activations (JARRETT et al., 2009).

We implemented the neural network using PyTorch (PASZKE et al., 2019) (version 1.2.0) and the `qpth` package (AMOS; KOLTER, 2017) (version 0.0.15) for the quadratic programming layer. We also used Scikit-Learn (PEDREGOSA et al., 2011) (version 0.21.3) and NumPy (OLIPHANT, 2006–) (version 1.17.3) to compute metrics, implement the shallow baselines and for miscellaneous tasks, and Torchmeta (DELEU et al., 2019) (version 1.1.1) to sample mini-batches of tasks, like described in Section 3.1. All our code is publicly available at https://github.com/gdahia/meta_occ/.

5.3.2 Optimization and Hyperparameters

We optimize both Meta SVDD and One-class Prototypical Networks using stochastic gradient descent (ROBBINS; MONRO, 1951) on the objective defined in Section 3.1 and Equation 3.4 with the Adam optimizer (KINGMA; BA, 2015). We use a constant learning rate of 5×10^{-4} over mini-batches of tasks of size 16, each having set X' with 5 examples, and set Z' with 10 examples from the target class and 10 examples from a randomly picked negative class. The learning rate value was the first one we tried, so no tuning was required.

We picked the task batch size that performed better in the validation set when training halts; we tried sizes $\{2, 4, \dots, 32\}$. We evaluate the performance in the validation set with 95% confidence intervals of the model’s accuracy. These confidence intervals are computed over 500 tasks, and we sample them randomly from the validation sets. To select the best model, we define that a model is better than another if the lower bound of its confidence interval is greater than that of the other. We break ties (*i.e.*, their lower bounds are equal up to 5 decimal points) by considering the one with higher mean accuracy to be better.

Early stopping halts training when performance in the validation set does not increase for 10 evaluations in a row. In all experiments, we use the model with higher performance in the validation set. We evaluate the model in the validation set every 100 training steps.

5.3.3 Baselines

The results for the few-shot experiment with Deep SVDD are obtained modifying the code made available by the authors¹, keeping the same hyperparameters.

For the few-shot baseline accuracy experiment with Principal Component Analysis (PCA) and One-Class Support Vector Machines (OCSVMs) with Gaussian kernel, we use the grid search space used by the experiments in prior work (RUFF et al., 2018): γ is selected from $\{2^{-10}, 2^{-9}, \dots, 2^{-1}\}$, and ν is selected from $\{0.01, 0.1\}$. Furthermore, we give the shallow baseline an advantage by evaluating every parameter combination in the test set and reporting the best result.

5.4 RESULTS

Table 5.1 Minimum, median and maximum mean AUC alongside their standard deviation for one-class classification methods for 10 repetitions. We highlight in boldface the highest mean and others which are within one standard deviation from it. The results for the many-shot baseline DCAE in MNIST and CIFAR-10 are compiled from the table by Ruff *et al.* (RUFF et al., 2018). The results for Omniglot and CIFAR-FS are for 5-shot one-class classification, and as noted before, it is unfeasible to report results for Deep SVDD in the Omniglot dataset.

	Dataset	DCAE	Deep SVDD	Dataset	Deep SVDD	One-Class Protonet	Meta SVDD
Min.	MNIST	78.2 ± 2.7	88.5 ± 0.9	Omniglot	-	89.0 ± 0.2	88.6 ± 0.4
Med.		86.7 ± 0.9	94.6 ± 0.9			99.5 ± 0.0	99.5 ± 0.0
Max.		98.3 ± 0.6	99.7 ± 0.1			100.0 ± 0.0	100.0 ± 0.0
Min.	CIFAR-10	51.2 ± 5.2	50.8 ± 0.8	CIFAR-FS	47.9 ± 4.9	60.2 ± 3.4	59.0 ± 5.7
Med.		58.6 ± 2.9	65.7 ± 2.5		64.0 ± 5.0	72.7 ± 3.0	71.0 ± 4.0
Max.		76.8 ± 1.4	75.9 ± 1.2		92.4 ± 2.3	90.1 ± 2.3	92.5 ± 1.7

5.4.1 First Experiment

We reproduce the results reported for Deep SVDD (RUFF et al., 2018) and its baselines alongside the results for 5-shot Meta SVDD and One-class Prototypical Networks, and our experiment with 5-shot Deep SVDD in Table 5.1. Figure 5.1 also provides mean AUC with shaded standard deviations for the results in the CIFAR dataset variants.

While the results from different datasets are not comparable due to the differences in setting and application listed in Section 5.1, they show that the approach has similar performance to the many-shot state-of-the-art in terms of AUC. Figure 5.1 shows that when we sort the mean AUCs for CIFAR-10 and CIFAR-FS, the performance from hardest

¹<https://github.com/lukasruff/Deep-SVDD-Pytorch>

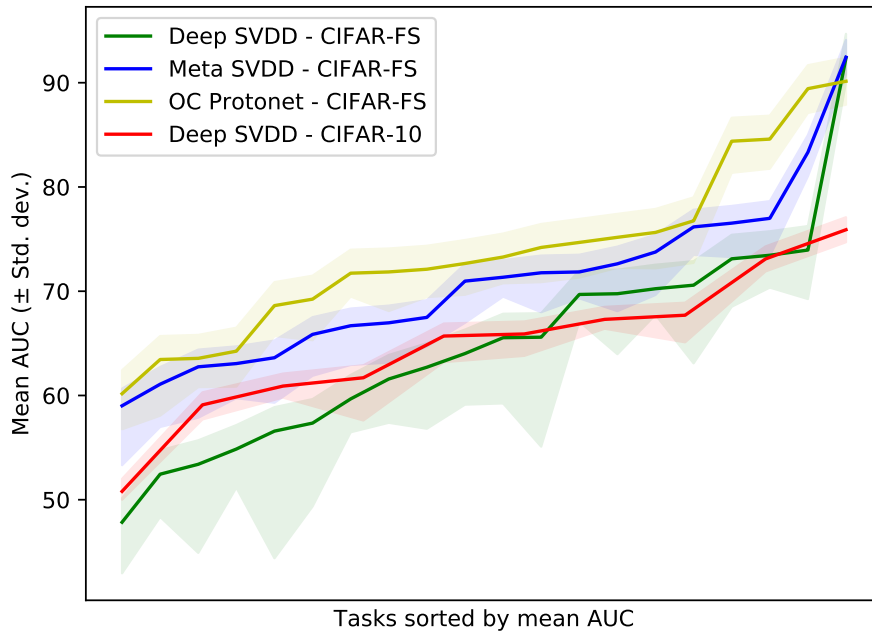


Figure 5.1 Mean AUC with shaded standard deviations for tasks in CIFAR datasets sorted by increasing mean value. Comparing Deep SVDD across datasets and protocols shows that the modified protocol is reasonable to evaluate few-shot one-class classification because the trend in task difficulty is similar. Within the few-shot protocol in CIFAR-FS, meta one-class classification are numerically superior, show less variance and can be meta-trained once for all tasks, with simple adaptation for unseen tasks, but require related task data.

to easier tasks exhibit similar trends despite these differences. These results also show that the modifications to the protocol are reasonable.

This experiment is evidence that our method is able to reduce the required amount of data from the target class in case we have labeled data from related tasks. Note that it is not the objective of our experiments to show that our method has better performance than previous approaches, since they operate in different settings, *i.e.* few-shot with related tasks and many-shot without them.

The comparison with Deep SVDD in the few-shot scenario gives further evidence of the relevance of our method: both Meta SVDD and One-Class Prototypical Networks obtain higher minimum, and median AUC than Deep SVDD. Another advantage is that we train f_θ once in the training set of Omniglot or CIFAR-FS, and learn only either the SVDD or the average on each of the sets X in the test set. We also obtain these results without any pre-training, and we have established a clear validation procedure to guide hyperparameter tuning and early stopping.

These results also show we can train a neural network for f_θ without architectural restrictions to optimize a one-class classification objective whereas other methods either require feature engineering, optimize another metric, or impose restrictions on the model architecture to prevent learning trivial functions.

Table 5.2 Mean accuracy alongside 95% confidence intervals computed over 10,000 tasks for Gaussian kernel One-class SVM with PCA, Meta SVDD and One-class Prototypical Networks. The results with highest mean and those with overlapping confidence interval with it are in boldface. We report the best result for the One-class SVM in its parameter search space, which gives it an advantage over the other two methods. Despite employing a simpler algorithm for one-class classification, One-class Prototypical networks obtain equivalent accuracy for Omniglot and better accuracy for CIFAR-FS than Meta SVDD. This indicates that learning feature representations is more important than which one-class classification algorithm we use.

Dataset	PCA+SVM	One-class Protonet	Meta SVDD
Omniglot	50.64 \pm 0.10%	94.68 \pm 0.17%	94.33 \pm 0.19%
CIFAR-FS	54.77 \pm 0.31%	67.67 \pm 0.39%	64.95 \pm 0.37%

The final advantage of our method over Deep SVDD is that the related tasks give predictive measures of metrics of interest like AUC, which allows tuning hyperparameters and using early stopping to regularize the training.

5.4.2 Second Experiment

The results for our second experiment, comparing the accuracies of Meta SVDD, a shallow baseline and One-class Prototypical Networks are presented in Table 5.2.

In this experiment, we can see an increase from almost random performance to almost perfect performance for both methods when compared to the shallow baseline in Omniglot. Both methods for few-shot one-class classification that use related tasks have equivalent performance in Omniglot. The gain is not as significant for CIFAR-FS but more than 10% in absolute for both methods, which shows they are a marked improvement over the shallow baseline.

We attribute the improvement to learning f_θ from related tasks, which addresses the feature engineering problem of OCSVM. Our method requires only a small number of examples to learn the one-class classification boundary, solving the scalability problem in the number of samples experienced by the shallow baseline. Finally, by making the feature dimension d much smaller than the original dimension of the input space D , we address the scalability issue regarding the feature dimensionality that affects both traditional SVDD and OCSVM.

Comparing the two proposed methods, we observe the unexpected result that the simpler method, One-class Prototypical Networks, has equivalent accuracy in the Omniglot experiment, and *better* accuracy in the CIFAR-FS experiment. This indicates that learning the feature representation directly from data might be more important than the one-class classification algorithm we choose, and the increased complexity of using SVDD over simple averaging does not translate into improved performance in this setting.

5.4.3 Other Experiments

We have also attempted to run this same experiment with the *miniImageNet* dataset (VINYALS et al., 2016), a dataset for few-shot learning using the images from the ImageNet dataset (DENG et al., 2009). The accuracy in the validation set, however, never rose above 50%. One of the motivations of introducing CIFAR-FS was that there was a gap in the challenge between training models in Omniglot and *miniImageNet* and that successfully training models in the latter took hours (BERTINETTO et al., 2019). Since none of the previous methods attempted solving ImageNet level datasets, and the worst performance in datasets from CIFAR is already near random guessing, we leave the problem of training one-class classification algorithms in this dataset open for future work.

We have run a small variation of the second experiment in which the number of examples in X is greater than during training, using 10 examples instead of 5. The results stayed within the accuracy confidence intervals for 5-shot for both models in this 10-shot deployment scenario.

Finally, we have also conducted experiments for One-class Prototypical networks, the most promising of the two methods, in a practical task: few-shot face verification.

FACE VERIFICATION EXPERIMENTS

In our experiments to evaluate our method in the few-shot face verification setting, we divide the training dataset into training and validation subsets (Section 6.3). Then, we use the entire training set to train a given method, using the validation set to tune hyperparameters and to select models (Section 6.4). It is important to remark we train and validate each technique according to its prescribed methodology; only evaluation in the test set must follow the few-shot face verification protocol.

After selecting both baseline and proposed models using the same procedure, we evaluate it using the few-shot protocol inspired by meta-learning few-shot classification in the test set and report traditional face recognition metrics (Section 6.2). We compare One-class prototypical networks, described in Section 3.3, with a strong and traditional baseline for face recognition, metric learning with triplet-loss (Section 6.1) (SCHROFF; KALENICHENKO; PHILBIN, 2015).

6.1 BASELINE METHOD

To assess the practical applicability of the method we propose, we compare it with a strong face recognition baseline, triplet-loss (SCHROFF; KALENICHENKO; PHILBIN, 2015). We choose triplet-loss because it is well established, its performance does not require much hyperparameter tuning and it has competitive or superior performance than more recent work (MUSGRAVE; BELONGIE; LIM, 2020b).

Triplet-loss is a loss function for metric learning: given face images alongside identity labels, first we convert these images into embeddings, and then we find triplets composed of anchor, positive and negative embeddings. The anchor and positive examples belong to the same identity, and the negative must belong to another identity. Learning then proceeds by updating the model’s parameters to minimize the number of triplets where the Euclidean distance between the negative example and the anchor is less than its distance to the positive example. As is usual with neural networks and deep learning, these updates are in the form of small steps following the direction of the negative gradient of the loss function.

The number of triplets in a dataset scales cubically with its size measured in number of examples, and so to make this objective feasible, triplet-loss is combined with heuristics to select triplets, which are called mining strategies. The most common of these, proposed by the same authors (SCHROFF; KALENICHENKO; PHILBIN, 2015), is semi-hard mini-batch mining: given a desired number of triplets, this mining strategy chooses triplets that maximize the margin between the negative and positive examples, and discards positive examples that are too close to the anchor already. This is the mining strategy we use, and we will refer to it hereon simply by triplet-loss.

Triplet-loss is designed for face recognition, but has since been adopted more widely for metric learning (MUSGRAVE; BELONGIE; LIM, 2020b). These uses, however, do not include few-shot face verification. Therefore, the comparison we make is unfair to triplet-loss, specially in the protocol we design: it only receives one image of the support face instead of the few our method obtain, and inference is done by computing the similarity score as the minus the distance between the query and support face embeddings. Where One-Class Prototypical networks get K support images and from them creates a model, Triplet-Loss gets only one image from the genuine identity, and makes no model from it. We could, for example, use the embeddings of the support set to normalize the Triplet-Loss embeddings, or compute their average pairwise distance and use this to normalize against imposters. So, in a way, we are comparing the paradigm of face recognition as is with the proposed few-shot face verification using our method.

6.2 EVALUATION PROTOCOL

To compare the methods, we combine the usual protocol of few-shot classification used to evaluate methods in meta-learning with traditional metrics used for face recognition and biometrics evaluation in general.

The meta-learning part of the protocol is that we separate the test set in episodes and use them to compute our metrics of interest, following standard practice in the field (BERTINETTO et al., 2019; LEE et al., 2019). Each episode consists of a support set and a query set. The examples in the support set all belong to a single identity, which we refer to as the target or genuine identity. In terms of few-shot face verification, these would be the examples the user provides to the system; they are therefore used to adapt the classifier to the given task.

The examples in the query set can be from any identity in the test set, including the target identity, and are meant to emulate the access to the system adapted to the target identity. The method hence performs better if it detects faces from an identity which is not the target as impostors, and the faces belonging to the target identity as genuines. Each support set contains 5 examples of the genuine identity, and each query set contains 5 distinct examples of 5 distinct identities, including the genuine. We also permute the episodes to avoid repeating them. Overall, we use 40,000 episodes, grouped in batches of size 4 for performance.

After computing scores for each example in the query set for all episodes, we summarize all of them into a single Receiver Operating Characteristic (ROC) curve for each method, and report the Equal Error Rate (EER) for each curve. This allows us to draw

conclusions even in the presence of more impostor queries than genuine ones, and follows standard practice in face recognition and biometrics research more broadly.

6.3 DATASET

We use the VGGFace2 dataset, described in Section 4.3, in our experiments, and we respect the training and testing splits provided by its authors (CAO et al., 2018). However, we further separate 20% of the identities randomly in the training set to become our validation set during training.

Furthermore, we pre-process the face images as follows:

1. we use the authors' provided bounding boxes to crop a central region over the face, keeping a margin of 32 pixels around it. By cropping the images using these fixed bounding boxes, we isolate the performance obtained in the experiments from face detection and normalization issues. This also improves reproducibility, by depending only on data available alongside the dataset and no further learned face detection module.
2. we resize the images to 128×128 pixels and compress them using JPEG. Doing this, we reduce the dataset size from hundreds of Gigabytes to tens of Gigabytes, and improve the runtime of the data input pipeline considerably.
3. We save the images in HDF5 format and the corresponding identity labels in JSON format.

6.4 SETUP

We conduct our face verification experiments using a Convolutional Neural Network (CNN) as the model to compute real valued, 256-dimension embeddings from face images. The CNN architecture we use is the Myrtle-5 non-residual model (PAGE, 2018; SHANKAR et al., 2020) because of its small number of parameters, simplicity and low computational cost. Despite not being a traditional network architecture for face recognition, thus obtaining performance much below what is currently expected of state-of-the-art face recognition methods, using the Myrtle-5 allows us to run experiments with many fewer computational resources, and we do not extend our conclusions to these other models. Furthermore, our intended application was on-device few-shot face verification, which would require such low-resource network architecture.

We follow standard practice and optimize the hyperparameters of neural network that generates the embeddings using mini-batch stochastic gradient descent, with the Adam (KINGMA; BA, 2015) update rule. For the One-class Protoypical Network model, we create mini-batches with 4 episodes, and each episode contains 5 identities (5-way) and each identity is represented by 5 images (5-shot). We reduce the number of episodes and identities compared to our previous experiments due to computational resource constraints.

For the baseline, triplet-loss model, we create standard mini-batches using the total number of identities and faces per identity to equal the number we use for our method,

which is 10 examples per identity and 20 identities per batch. The learning rate for both methods was set manually by observing the validation metrics over the first 5 validation epochs.

We use the loss function of each method unchanged, and use the validation set with the metrics appropriate to each method - *i.e.* mean accuracy over episodes for One Class Prototypical Network, and validation loss for Triplet Loss - to select models based on early-stopping with a tolerance of 5 non-improving rounds of 1,000 batches. We then select the best model obtained using this procedure for further evaluation.

We implement both methods using Pytorch (PASZKE et al., 2019), and the auxiliary Torchmeta (DELEU et al., 2019) and Pytorch Metric Learning (MUSGRAVE; BELONGIE; LIM, 2020a) libraries for the One-class Prototypical Network and Triplet-loss implementations, respectively. We use the standard hyperparameters for these libraries regarding the loss functions.

6.5 RESULTS

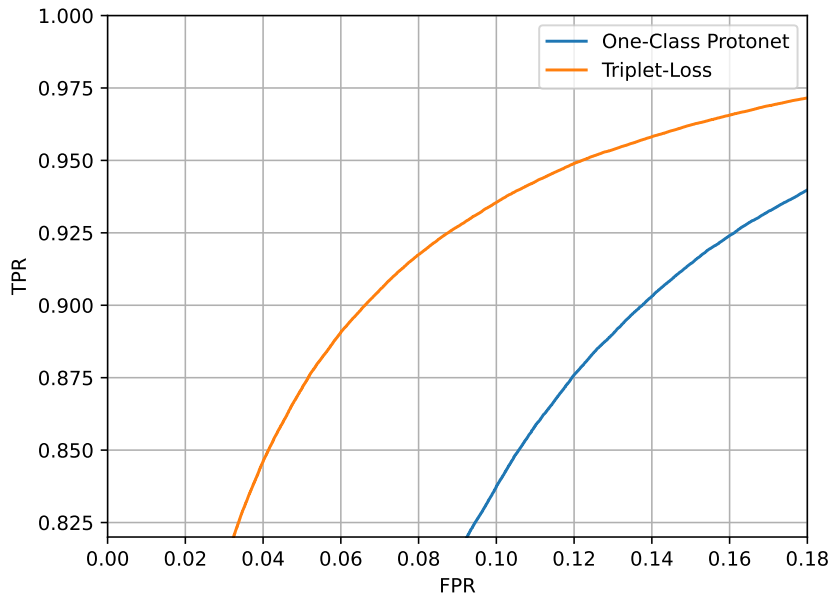


Figure 6.1 ROC curve for the face verification experiment. Notice that the figure depicts only the upper-left corner of the curve – from 0.82 to 1 in the TPR axis and from 0 to 0.2 in the FPR axis – to make the comparison between both methods easier to analyze. The one-class prototypical network, when used for few-shot face verification, attains a lower value of TPR than the triplet-loss baseline if both operate at the same FPR for the entire displayed area of the graph. This happens even in this setting where Triplet loss does not use the support set fully, and is therefore in a disadvantage.

Table 6.1 EERs for the triplet loss baseline and the one-class prototypical network methods in the face verification experiment in VGGFace2. Lower values are better, hence the baseline triplet-loss is superior to the proposed method in this experiment even in a disadvantageous setting.

	One-class Protonet	Triplet-loss
EER	12.19%	8.05%

Figure 6.1 shows the ROC curve for the face verification experiment, and Table 6.1 displays the EER obtained for the One-Class Protonet and the Triplet-loss baseline in the same experiment. We can observe that the baseline, despite having access only to one support example, outperforms the proposed method in the entire displayed range and attains a lower EER; we imagine that adapting it to fully leverage the support set could increase its advantage.

The only advantage we perceive of Triplet-Loss over the proposed methods is that it has access to all labels during optimization of the network, whereas the meta one-class classification methods we propose require only pairwise labels indicating whether two elements belong to the same class or not.

As the One-Class Prototypical network obtained better results than the Meta Support Vector Data Description (SVDD) method in the validation experiments (Chapter 5), it is easy to infer that it would also underperform the Triplet-Loss baseline.

It is also worth noting that the results we obtained from these experiments are not on the level of the current state-of-the-art of face recognition in the standard benchmarks (HUANG et al., 2007; TAIGMAN et al., 2014; CAO et al., 2018; WANG; DENG, 2021), including for the baseline, where we do not reproduce the original results (SCHROFF; KALENICHENKO; PHILBIN, 2015). We attribute this to the smaller size of the neural network architectures we use, which despite allowing us to reach the above conclusions with reduced computational power, prevents us from comparing these results with that of methods studied in other work.

CONCLUSION

7.1 META-ONE CLASS CLASSIFICATION

We have described a way to learn feature representations so one-class classification algorithms can learn decision boundaries that contain the target class from data, optimizing an estimator of its true objective. Furthermore, this method works with 5 samples from the target class with performance similar to the state-of-the-art in the setting where target class data is abundant, and better when the many-shot state-of-the-art method is employed in the few-shot setting. We also provide an experiment that shows that using a simpler one-class classification yields comparable performance, displaying the advantages of learning feature representations directly from data.

One possibility to replace the main requirement of our method with a less limiting one would be the capability of generating related tasks from unlabeled data. A simple approach in this direction could be using weaker learners to define pseudolabels for the data. Doing this successfully would increase the number of settings where our method can be used significantly.

The main limitations of our method besides the requirement of the related tasks are the destabilization of the quadratic programming layer, which we solved by adding a stabilization term to the diagonal of the kernel matrix or by simplifying the one-class classification algorithm to use the mean of the features, and its failure to obtain meaningful results in the *mini*ImageNet dataset.

We believe not only finding solutions to these limitations should be investigated in future work but also other questions left open in our work, like confirming our hypothesis that introducing slacks would not benefit Meta Support Vector Data Description (SVDD).

Other directions for future work are extending our method for other settings and using other one-class classification methods besides SVDD. Tax and Duin (TAX; DUIN, 2004) also detail a way to incorporate negative examples in the SVDD objective, so we could try learning f_θ using this method and to minimize the hypersphere's volume instead of converting SVDD into a binary classification problem that uses the unseen examples' distances to the center as logits. Another avenue for future work is that we could avoid

using labeled related tasks by using only examples created using mixup (ZHANG et al., 2018) from examples in the positive class.

7.2 FEW-SHOT FACE VERIFICATION

With our face verification experiments, we found that the results for the proposed meta one-class classification methods did not improve over the well-established baseline even when comparing to it in an unfavorable scenario. We believe this could indicate that the meta-learning community could instead learn more from metric learning and face recognition work, instead of otherwise, and techniques like triplet mining could improve the results even if lacking some of the nice theoretical underpinnings of current meta-learning techniques. Recent work already notes that, for example, the division of mini-batches into episodes or tasks is harmful to prototypical networks in few-shot classification (LAENEN; BERTINETTO, 2021). We leave the investigation of the performance of our method without this division as future work.

Furthermore, as we noted, the triplet-loss baseline leverages the entirety of mini-batch labels during training, whereas the methods we proposed for meta one-class classification require only pairwise comparison labels for the examples. Perhaps our methods could be useful in other practical applications, like credit card fraud, where imposter buy attempts are not associated with a fixed identity and cannot be compared to identities of other credit card users.

This also shows another avenue for future work: there seems to be other, useful applications for meta one-class classification where metric-learning is not suitable, and where our method could be advantageous.

BIBLIOGRAPHY

- AMOS, B.; KOLTER, J. Z. OptNet: Differentiable optimization as a layer in neural networks. In: PRECUP, D.; TEH, Y. W. (Ed.). *Proceedings of the 34th International Conference on Machine Learning*. International Convention Centre, Sydney, Australia: PMLR, 2017. (Proceedings of Machine Learning Research, v. 70), p. 136–145. Available: [⟨http://proceedings.mlr.press/v70/amos17a.html⟩](http://proceedings.mlr.press/v70/amos17a.html).
- ARJOVSKY, M.; CHINTALA, S.; BOTTOU, L. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.
- BERTINETTO, L. et al. Meta-learning with differentiable closed-form solvers. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. [s.n.], 2019. Available: [⟨https://openreview.net/forum?id=HyxnZh0ct7⟩](https://openreview.net/forum?id=HyxnZh0ct7).
- CAGLE, M. *California Just Blocked Police Body Cam Use of Face Recognition*. [S.l.]: American Civil Liberties Union, 2019. [⟨https://www.aclu.org/blog/privacy-technology/surveillance-technologies/california-just-blocked-police-body-cam-use-face⟩](https://www.aclu.org/blog/privacy-technology/surveillance-technologies/california-just-blocked-police-body-cam-use-face). Accessed: 2021-9-1.
- CAO, Q. et al. VGGFace2: A dataset for recognising faces across pose and age. In: *International Conference on Automatic Face and Gesture Recognition*. [S.l.: s.n.], 2018.
- CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, v. 20, n. 3, p. 273–297, 1995.
- DELEU, T. et al. *Torchmeta: A Meta-Learning library for PyTorch*. 2019. Available at: [⟨https://github.com/tristandeleu/pytorch-meta⟩](https://github.com/tristandeleu/pytorch-meta). Available: [⟨https://arxiv.org/abs/1909.06576⟩](https://arxiv.org/abs/1909.06576).
- DENG, J. et al. Imagenet: A large-scale hierarchical image database. In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. [S.l.: s.n.], 2009. p. 248–255.
- DENG, J. et al. Arcface: Additive angular margin loss for deep face recognition. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2019. p. 4690–4699.
- ELZINGA, D. J.; HEARN, D. W. The minimum covering sphere problem. *Management science*, INFORMS, v. 19, n. 1, p. 96–104, 1972.

ERFANI, S. M. et al. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, v. 58, p. 121 – 134, 2016. ISSN 0031-3203. Available: [⟨http://www.sciencedirect.com/science/article/pii/S0031320316300267⟩](http://www.sciencedirect.com/science/article/pii/S0031320316300267).

FINN, C.; ABBEEL, P.; LEVINE, S. Model-agnostic meta-learning for fast adaptation of deep networks. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. [S.l.]: JMLR.org, 2017. (ICML'17), p. 1126–1135.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. [⟨http://www.deeplearningbook.org⟩](http://www.deeplearningbook.org).

GOODFELLOW, I. J. et al. Generative adversarial nets. In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. [s.n.], 2014. p. 2672–2680. Available: [⟨http://papers.nips.cc/paper/5423-generative-adversarial-nets⟩](http://papers.nips.cc/paper/5423-generative-adversarial-nets).

GUO, Y.; ZHANG, L. One-shot face recognition by promoting underrepresented classes. *arXiv preprint arXiv:1707.05574*, 2017.

GUO, Y. et al. MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*. [s.n.], 2016. p. 87–102. Available: [⟨https://doi.org/10.1007/978-3-319-46487-9_6⟩](https://doi.org/10.1007/978-3-319-46487-9_6).

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. *Science*, American Association for the Advancement of Science, v. 313, n. 5786, p. 504–507, 2006. ISSN 0036-8075. Available: [⟨https://science.sciencemag.org/content/313/5786/504⟩](https://science.sciencemag.org/content/313/5786/504).

HUANG, G. B. et al. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. [S.l.], 2007.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. [s.n.], 2015. p. 448–456. Available: [⟨http://proceedings.mlr.press/v37/ioffe15.html⟩](http://proceedings.mlr.press/v37/ioffe15.html).

JARRETT, K. et al. What is the best multi-stage architecture for object recognition? In: *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*. [s.n.], 2009. p. 2146–2153. Available: [⟨https://doi.org/10.1109/ICCV.2009.5459469⟩](https://doi.org/10.1109/ICCV.2009.5459469).

KEMELMACHER-SHLIZERMAN, I. et al. The megaface benchmark: 1 million faces for recognition at scale. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2016. p. 4873–4882.

- KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. [s.n.], 2015. Available: [⟨http://arxiv.org/abs/1412.6980⟩](http://arxiv.org/abs/1412.6980).
- KLARE, B. F. et al. Pushing the frontiers of unconstrained face detection and recognition: Iarpa Janus benchmark A. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 1931–1939.
- KRIZHEVSKY, A. *Learning multiple layers of features from tiny images*. [S.l.], 2009.
- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105.
- LAENEN, S.; BERTINETTO, L. *On Episodes, Prototypical Networks, and Few-shot Learning*. 2021.
- LAKE, B. M.; SALAKHUTDINOV, R.; TENENBAUM, J. B. Human-level concept learning through probabilistic program induction. *Science*, American Association for the Advancement of Science, v. 350, n. 6266, p. 1332–1338, 2015. ISSN 0036-8075. Available: [⟨https://science.sciencemag.org/content/350/6266/1332⟩](https://science.sciencemag.org/content/350/6266/1332).
- LECUN, Y. et al. Handwritten digit recognition with a back-propagation network. In: *Advances in neural information processing systems*. [S.l.: s.n.], 1990. p. 396–404.
- LeCun, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998. ISSN 1558-2256.
- LEE, K. et al. Meta-learning with differentiable convex optimization. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2019.
- MESCHEDER, L.; GEIGER, A.; NOWOZIN, S. Which training methods for GANs do actually converge? In: DY, J.; KRAUSE, A. (Ed.). *Proceedings of the 35th International Conference on Machine Learning*. Stockholmssäsan, Stockholm Sweden: PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 3481–3490. Available: [⟨http://proceedings.mlr.press/v80/mescheder18a.html⟩](http://proceedings.mlr.press/v80/mescheder18a.html).
- MURGIA, M. *Microsoft quietly deletes largest public face recognition data set*. 2019. Accessed: 2021-9-1. Available: [⟨https://www.ft.com/content/7d3e0d6a-87a0-11e9-a028-86cea8523dc2⟩](https://www.ft.com/content/7d3e0d6a-87a0-11e9-a028-86cea8523dc2).
- MURGIA, M. *Microsoft quietly deletes largest public face recognition data set*. 2019. Accessed: 2021-9-1. Available: [⟨https://www.ft.com/content/7d3e0d6a-87a0-11e9-a028-86cea8523dc2⟩](https://www.ft.com/content/7d3e0d6a-87a0-11e9-a028-86cea8523dc2).
- MUSGRAVE, K.; BELONGIE, S.; LIM, S.-N. *PyTorch Metric Learning*. 2020.

- MUSGRAVE, K.; BELONGIE, S. J.; LIM, S. A metric learning reality check. *CoRR*, abs/2003.08505, 2020. Available: [⟨https://arxiv.org/abs/2003.08505⟩](https://arxiv.org/abs/2003.08505).
- OLADOSU, A. et al. Meta-learning for anomaly classification with set equivariant networks: Application in the milky way. *arXiv preprint arXiv:2007.04459*, 2020.
- OLIPHANT, T. *NumPy: A guide to NumPy*. 2006–. USA: Trelgol Publishing. Available: [⟨http://www.numpy.org/⟩](http://www.numpy.org/).
- OZA, P.; PATEL, V. M. One-class convolutional neural network. *IEEE Signal Process. Lett.*, v. 26, n. 2, p. 277–281, 2019.
- PAGE, D. *myrtle.ai*. 2018. Accessed: 2021-12-14. Available: [⟨https://myrtle.ai/how-to-train-your-resnet-4-architecture/⟩](https://myrtle.ai/how-to-train-your-resnet-4-architecture/).
- PASZKE, A. et al. Pytorch: An imperative style, high-performance deep learning library. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019. p. 8024–8035. Available: [⟨http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf⟩](http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf).
- PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.
- PERERA, P.; OZA, P.; PATEL, V. M. *One-Class Classification: A Survey*. 2021.
- PIMEYES. *PimEyes: Face Recognition Search Engine and Reverse Image Search*. 2017. [⟨https://pimeyes.com/en⟩](https://pimeyes.com/en). Accessed: 2021-9-1.
- RAGHU, A. et al. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. *CoRR*, abs/1909.09157, 2019. Available: [⟨http://arxiv.org/abs/1909.09157⟩](http://arxiv.org/abs/1909.09157).
- RAJI, I. D. et al. Saving face: Investigating the ethical concerns of facial recognition auditing. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. New York, NY, USA: Association for Computing Machinery, 2020. (AIES '20), p. 145–151.
- ROBBINS, H.; MONRO, S. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, v. 22, n. 3, p. 400–407, set. 1951. ISSN 0003-4851, 2168-8990. Available: [⟨https://projecteuclid.org/euclid.aoms/1177729586⟩](https://projecteuclid.org/euclid.aoms/1177729586).
- RUFF, L. et al. Deep one-class classification. In: DY, J.; KRAUSE, A. (Ed.). *Proceedings of the 35th International Conference on Machine Learning*. Stockholmsmässan, Stockholm Sweden: PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 4393–4402. Available: [⟨http://proceedings.mlr.press/v80/ruff18a.html⟩](http://proceedings.mlr.press/v80/ruff18a.html).
- SCHLEGL, T. et al. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: *Information Processing in Medical Imaging - 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings*. [S.l.: s.n.], 2017. p. 146–157.

- SCHÖLKOPF, B. et al. Estimating the support of a high-dimensional distribution. *Neural Computation*, v. 13, n. 7, p. 1443–1471, 2001.
- SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 815–823.
- SEEBÖCK, P. et al. Identifying and categorizing anomalies in retinal imaging data. *CoRR*, abs/1612.00686, 2016. Available: [⟨http://arxiv.org/abs/1612.00686⟩](http://arxiv.org/abs/1612.00686).
- SHANKAR, V. et al. Neural kernels without tangents. mar. 2020. Available: [⟨http://arxiv.org/abs/2003.02237⟩](http://arxiv.org/abs/2003.02237).
- SNELL, J.; SWERSKY, K.; ZEMEL, R. S. Prototypical networks for few-shot learning. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. [S.l.: s.n.], 2017. p. 4077–4087.
- TAIGMAN, Y. et al. DeepFace: Closing the gap to human-level performance in face verification. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2014. p. 1701–1708.
- TAX, D. M. J.; DUIN, R. P. W. Support vector data description. *Machine Learning*, v. 54, n. 1, p. 45–66, 2004.
- VINYALS, O. et al. Matching networks for one shot learning. In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. [s.n.], 2016. p. 3630–3638. Available: [⟨http://papers.nips.cc/paper/6385-matching-networks-for-one-shot-learning⟩](http://papers.nips.cc/paper/6385-matching-networks-for-one-shot-learning).
- WANG, H. et al. Cosface: Large margin cosine loss for deep face recognition. *CoRR*, abs/1801.09414, 2018. Available: [⟨http://arxiv.org/abs/1801.09414⟩](http://arxiv.org/abs/1801.09414).
- WANG, M.; DENG, W. Deep face recognition: A survey. *Neurocomputing*, v. 429, p. 215–244, 2021. ISSN 0925-2312. Available: [⟨https://www.sciencedirect.com/science/article/pii/S0925231220316945⟩](https://www.sciencedirect.com/science/article/pii/S0925231220316945).
- WEN, Y. et al. A discriminative feature learning approach for deep face recognition. In: SPRINGER. *European conference on computer vision*. [S.l.], 2016. p. 499–515.
- Wolf, L.; Hassner, T.; Maoz, I. Face recognition in unconstrained videos with matched background similarity. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2011. p. 529–534. ISSN 1063-6919.
- ZHANG, H. et al. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2018.