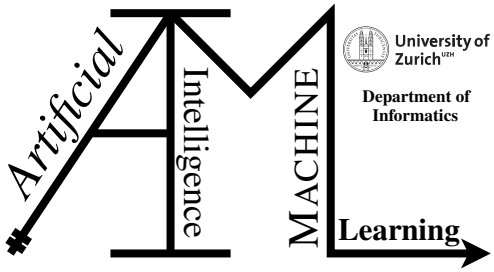
Deep Radial Basis-Function Networks for Open-Set Classification

Master's Thesis

Remo Hertig
13-738-323

Submitted on
September 2, 2023

Thesis Supervisor
Prof. Dr. Manuel Günther



Master's Thesis

Author: Remo Hertig, remotobias.hertig@uzh.ch

Project period: March 2, 2023 - September 2, 2023

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

Acknowledgements

I would like to express my deepest gratitude to Prof. Dr. Manuel Günther for his valuable advice, unwavering guidance and the numerous fruitful discussions. I would also like to extend my appreciation to my family and friends for their encouragement and unconditional support.

Abstract

A problem with modern deep learning recognition systems is that they often respond to stimuli of an unknown class overly confident, but wrong. Open-set recognition highlights this behavior and provides evaluation methods to estimate the generalization capability of models beyond the classic train/test set split. In this thesis, we incorporate a Radial Basis Function (RBF) layer into deep convolutional networks to model the deep feature distribution. We evaluate such networks on standard open-set evaluation protocols and compare their performance with standard Softmax classification models. Additionally, we utilize negative training samples and compare with the Entropic Open-Set Loss Softmax extension. We show that standard deep RBF network with Gaussian activation functions does not outperform Softmax based methods in open-set recognition. We extend the RBF network in two ways, which both show increased open-set recognition performance over the baseline RBF network. Based on these results we conjecture that solely using an RBF layer for the classification sub-system of a deep neural network might not be sufficient to solve the open-set recognition problem.

Contents

Chapter 1

Introduction

Turing (1950) posed the intriguing question of whether machines can think. One important constituent of thinking is the recognition of patterns (Minsky, 1961). In humans, recognition is an abstract capability operating on multiple modalities, tightly integrated with cognition. In general, it can be decomposed into three parts, first the identification of something familiar, second the recollection of various associated properties and third, naming it. In machines, all of these parts have to be studied, modeled and built explicitly. Naturally, a computational formalism which shares similarities with human thinking is rather appealing for this endeavor, such as artificial neural networks (Rosenblatt, 1958; Steinbuch, 1961). Initially there was quite some euphoria about these potential thinking machines, but it turned out that getting them to learn and solve moderately complex problems is far from trivial (Minsky and Papert, 1969). Thereafter, artificial neural networks had quite a shadowy existence for some time. Fortunately, many techniques were developed over the subsequent decades to get these new neural networks to learn complex problems (Schmidhuber, 2015). At the beginning of the century, advances in Graphics Processing Unit (GPU) technology provided massive speedups in parallel computation. This allowed for a renaissance in neural networks. They were adapted to the newly available accelerators, by the 2010s the so-called *Deep Learning Era* began, where deep learning started to surpass traditional machine learning methods (Sevilla et al., 2022). Not only did deep learning overtake traditional machine learning, but task by task, human performance was reached and surpassed. For example in image classification, measured on the ImageNet task (Russakovsky et al., 2015) a deep learning model surpassed the performance of a human expert for the first time in 2015 (He et al., 2015b). While these results are impressively showing the capabilities of neural networks, it is important to note that these tasks are usually evaluated under very carefully controlled laboratory conditions. Most importantly, it is assumed that there exists a fixed number of classes, which is known and that the dataset used contains samples from each of those classes. These assumptions have been termed *closed-set* (Majewski and Basztura, 1984). However, outside of the laboratory these assumptions no longer hold in most cases. A visual object recognition systems will not be constrained to inputs sampled from a set of predefined classes, but it could be exposed to any visual scene occurring "in the wild". Common deep-learning based classification systems will assign a class to any input, even if the input is completely unrelated due to the *close-world* assumption. To make matters worse, if the classifier's output is interpreted as a confidence of its decision, for example by treating the outputs as a probability after applying the well known *Softmax function*, it is very likely that the unrelated object is classified as a random class with a very high confidence (Dhamija et al., 2020). The impressive performance in the laboratory compared to the abysmal performance in the wild does raise serious questions on the applicability of the current approach. This issue is anything but new, in fact it has been known since the beginnings of the field.

With the arrival of the first commercial computers in the 1950s, research on artificial intelligence started formally and with it automated pattern recognition. It is plausible that these early computers have been the driving force behind the development of pattern recognition. Computers promised a huge increase in efficiency for businesses, especially when combined with efficient interfaces to the real world. A first important technology was the development of optical character recognition (Mori et al., 1992), which is a form of automated pattern recognition. During that time research on pattern recognition proliferated greatly and combined discoveries from various other branches of science (Nagy, 1968). A pattern recognition system was described by Kanal (1974) as a three stage process: measurement, feature extraction and classification. Measurement refers to the digitization of signals *e. g.* a camera in Optical Character Recognition (OCR). Feature extraction is the complex task of finding and selecting those bits of information of a measurement that would allow discriminating different patterns. Additionally, feature extraction reduces the data size and thus makes pattern recognition tractable in the first place on limited computing power. Classification refers to assigning each observation to one of many predefined classes (Nagy, 1968). In classification, number of possible classes was assumed to be fixed and known.Motivated by practical issues in OCR, such as errors in scanning or poor paper quality, Chow (1957) described a recognition system which can abstain from making a classification if the estimated error is too large. This type of system has been subsequently studied more thoroughly in (Chow, 1970). Thus already early on, the notion of *uncertainty* regarding the output of a pattern recognition systems was developed.

Uncertainy can be decomposed into two parts: "statistical" and "systematic" uncertainty (Hüllermeier and Waegeman, 2021). "Statistical uncertainty" is due to random effects in the data acquisition and is also known as *aleatoric* uncertainty. "Systematic" uncertainty originates from a mispecified model due to incomplete knowledge and is usually termed *epistemic* uncertainty. The method of Chow (1957) dealt with *aleatoric* uncertainty, because the model for characters occurring in a specific language is well defined. However, that author did also suggest to study *epistemic* uncertainty. Hellman (1970) did so by studying a nearest neighbor classification rule with a reject option. This system does not assume any knowledge of the statistics of the data, but only looks at the available labeled training data. If there is too much disagreement among the nearest neighbors about which class to assign to a new data point, it can be rejected as not belonging to any of the known ones. Thus this model does take *epistemic* uncertainty into account. Another type of *epistemic* uncertainty has been studied by Ide and Tunis (1967). They analyzed the robustness of a speech recognition system where the data distribution changes over time and propose an unsupervised method to account for the change. This effect on a classifier has been studied formally much later (Shimodaira, 2000) under the term *covariate shift*. Although the issue of the uncertainty of a pattern recognition systems has been known for quite some time (Duda, 1970), its systematic analysis has been somewhat ignored for some decades according to Herbei and Wegkamp (2006).

Dealing with these uncertainties is closely related to the notions of *generalization* (Bishop, 1995, p. 377ff), which is the capability of a recognition systems to make correct decisions on unseen data. However, evaluating this capability is a rather tricky endeavor. In general, the standard evaluation method for pattern recognition is the so called *test set error* or "accuracy", which is in turn dependent on the *test set*. The advantage is that it is easy to compute and there are strong theoretical justifications for this measure. Most statistical pattern recognition systems are grounded in *statistical learning theory*, which allow claims about *generalization* in the first place. Therein, a core assumption is that when the data used to *train and test* a system follows the same distribution as the one encountered in practice, then the average *test set error* becomes a good estimate of a certain notion of *generalization* (Devroye et al., 1997, p. 2ff). Great difficulties arise, when it is simply practically infeasible to construct such a dataset. This infeasibility is inherent in certain pattern recognition sub-fields, such as in *speaker recognition* (Majewski and Basztura, 1984) or *face*

recognition (Fayin Li and Wechsler, 2005).

As a first remedy, the classifier needs to be quipped with the capability to give a "don't know" answer, also known as *classification with rejection* (Chow, 1957; Hellman, 1970) and many more terms in different fields. While there can be multiple causes that should lead to a rejection decision in a classifier, we are mainly interested in the case where the inputs are visually similar to the training data, but very different semantically. Which happens to all machine learning system that were not trained data sampled from all possible input classes. This specific setting has been called *open-set* because data that is similar in the raw representation such as images our sounds (e.g voices), but semantically different (Majewski and Basztura, 1984). This *open-set recognition* terminology (Scheirer et al., 2013) is used in this work, because it contrasts nicely with the *closed-set* baseline.

A potential candidate classifier that incorporates the *open space risk* could be classic Radial Basis Function Networks (RBFN). Therein the classification output is determined by the distance of a sample to some "representation", which allow a probabilistic interpretation of class membership, including a "don't know" option if the distance of a sample to all radial basis centers is larger than some threshold. Unfortunately, shallow RBFN are notoriously difficult to train especially on large problems like ImageNet classification. However, RBFN are just one type of architecture and Radial Basis Functions could be applied in other manners.

While the *open set* recognition problem is of general nature, concerning all types of classifiers, recently it has been applied to classifiers constructed using deep learning (Bendale and Boult, 2016). Therein, the classifier has been equipped with a "don't know" option by estimating the likelihood that a sample belongs to any of the learned classes using the distribution of deep features and Extreme Value Theory. Their estimation of the "don't know" option is closely related to applying a Radial Basis Function with a Weibull kernel onto the deep features, but they are not using this information to inform the actual classifier, instead it's used as a so-called "Meta-Recognition" (Bendale and Boult, 2016). In the ontology of Hendrickx et al. (2021), the Extreme Value Theory approach is a *dependant rejector*, which can be used to extend other classifier models with an rejection option. Alternatively, an *integrated rejector* can be designed, where a single model performs the classification and rejection. While this is non-trivial, it has the theoretical advantage that both aspects can be optimized jointly, leading to better performance and less bias (Hendrickx et al., 2021).

Such an approach is the topic of this thesis. With the main question of whether such an integrated classification-rejection does in fact lead to better *open-set recognition* performance. To study this question, we first have to integrate *Radial Basis Functions* (RBF) into a deep learning network.

1.1 Research Questions

RQ1: Deep RBF Survey What does the literature tell us about parametrization, topology and interpretation of RBF neural networks?

RBFs appear in neural network at multiple places and in different forms. The goal of this question is to provide an overview of the design space of RBF neural networks with a focus on deep neural networks.

RQ2: Deep OSR What are the confounding factors in the open set recognition problem for deep neural networks?

Open set recognition with classifiers based on deep neural networks becomes increasingly difficult for unknowns that are more similar to the known training set, even with large datasets (Palechor et al., 2023). It is not obvious why that is the case. To answer this research

question, existing explanations and hints from the literature should be collected, critically analyzed and combined to provide an account of the core issue.

RQ3: Deep RBF prototypes What is the open set recognition performance for "prototypical" deep RBF networks with one center per known class?

Most recognition approaches are based on prototype theory. In this setting, it is assumed that a prototype exists which represents a condensed representation of all required features that explain class membership. Deep RBF networks provide an intuitive framework to model such an approach and offers fine-control on the representation. Such a network should be evaluated using common evaluation protocols (Bisgin et al., 2023) to provide a reference point for a baseline deep RBF variant. Additionally, the RBF should be compared to a standard Softmax CNN with Entropic Open-Set Loss (Dhamija et al., 2018).

RQ3.1: What loss function is appropriate for a RBF classification layer?

RQ3.2: What is the baseline performance for the handwriting recognition problem without using negative samples?

RQ3.3: What is the baseline performance for the handwriting recognition problem with using negative samples?

RQ3.4: What is the baseline performance for the image classification problem without using negative samples?

RQ3.5: What is the baseline performance for the image classification problem with using negative samples?

RQ4: Multiple Representations How does an exemplar approach compare with prototypes in OSR?

Most recognition approaches, including those described in the previous questions are based on prototype theory. In this setting, it is assumed that a prototype exists which represents a condensed representation of all required features that explain class membership.

An alternative view is the exemplar approach, therein classification is performed by similarity to the most similar *example* in memory. To adapt a classifier to work in the *exemplar* regime, it needs to allow for multiple centers per class. Presumably also the deep representation dimensions should be higher than in the prototype case.

RQ4.1: How to train with multiple representations?

RQ4.2: How does the performance differ compared to a prototype approach?
For comparison, the same experiment setup as in the *prototype* variant is used.

RQ5: Deep Local Metric Is it beneficial for deep representations to collapse at their RBF center?

Minimizing a loss using Gaussian RBF units will minimize their scale parameter σ. Intuitively this is useful because by minimizing within-class variance and maximizing between-class variance, a minimum description length could be achieved in the representation space, as a proxy for optimal representation of the deep features.

However, one might argue that this violates the *positivity* assumption of *metric spaces*, which states that distinct points have positive distance. If the deep feature representations are interpreted as a metric space, points from distinct samples should not have a distance of 0. It is not clear if this is of any practical relevance.

RQ5.1: What alternative functions could be used as activation function for the RBF such that representation collapse is prevented?

RQ5.2: Does such a RBF improve *open-set recognition* performance?

1.2 Notation

Throughout this work the following notation is used, unless otherwise indicated.

General

$\mathcal{J}$ loss function

$p(\cdot)$ probability density function

$\hat{p}(\cdot)$ estimated probability density function

y_c classification output / estimated probability for class c

t target output

Open-set recognition

$\mathcal{D}_p$ positive samples from known classes

$\mathcal{D}_n$ negative samples

$\mathcal{D}_u$ unknown test samples

Radial Basis Functions

$\phi(\cdot)$ radial basis function

$\delta(\cdot)$ distance function

$\psi(\cdot)$ activation function

1.3 Structure

The remainder of the thesis follows the structure: after a brief survey of works related to ours in Chapter 2, the necessary background on *radial basis functions* and *open-set recognition* is explained in more detail in Chapter 3. The datasets used in our experiments are described in Chapter 4. A brief exposition of *deep radial basis function neural networks* and an analysis of the *open-set recognition* problem is given in Chapter 5. Our chosen approach is described in Chapter 6 and the performed experiments in Chapter 7. The results are present in Chapter 8 and discussed in Chapter 9. We conclude the thesis in Chapter 10.

Chapter 2

Related Work

The main topic of this thesis is an investigation of a mathematical construct *Radial-Basis Functions* as a component in *statistical machine learning* evaluated in terms of *open-set recognition* performance. All of these three core concepts have a long history and quite some overlap. For the sake of brevity, only works related to the first and third will be briefly discussed in this section.

2.1 Open-set recognition

In this work *open-set recognition* refers to a viewpoint that the performance of a statistical recognition system can not be solely measured by the accuracy on the traditional train/test split. Because the possible inputs to the system outside of carefully controlled laboratory conditions are huge and not necessarily covered completely by the training data.

2.1.1 Related fields

Modern machine learning is not well known for having a standardized terminology and nomenclature nor for rigorously adhering to previously introduced terminology. This can lead to a lot of confusion and duplication of research effort. Sometimes terms are also deprived of their historical meaning to accommodate researchers instead of the other way around. Normally in science, whenever there is evidence that a certain characteristic has been neglected in previous research or a new and large sub problem is discovered, then there is a chance for a new sub-field to form. However, particularly in machine learning, in absence of nomenclature, whenever two terms are used to refer to the same thing, there is now a chance for at least two new sub-fields to form. With the result that it becomes increasingly difficult to disentangle all the subfields over time. Nonetheless, an attempt is made in this section to list research that is in some sense related with *open-set recognition*. Later on, *open-set recognition* itself will be discussed in more detail in Section 3.3.

Outliers

In statistics, the term *outlier* refers to an observation which is markedly different compared to some notion of "normal" data (Edgeworth, 1887). In the early days, such observations have been called *discordant observations*. *Outliers* have been a subject of study in statistics for centuries, particularly methods to identify them and how to deal with them (Beckman and Cook, 1983).

Outlier detection The field of *outlier detection* is concerned precisely with the detection of such *discordant observations* (Hodge and Austin, 2004). According to these authors, *anomaly detection* and *novelty detection* are synonyms for *outlier detection*.

Novelty detection According to Pimentel et al. (2014), *novelty detection* is concerned with detecting whether some new data is different from some other data. This in turn requires some sort of agreement on what "normal" data looks like and how different or *discordant observations* can be identified. Again, according to these authors, *anomaly detection* and *outlier detection* are not only synonyms for *novelty detection*. But all three of them are applications of *one-class classification*.

Anomaly detection Finding observations which are very different from others, like an outlier is a problem called *anomaly detection* according to Chandola et al. (2009). Unsurprisingly, *outlier detection* is again stated as a synonym for *anomaly detection*, such is *novelty detection*, although they differ in their motivation for application.

Out-of-Distribution detection A recently introduced, but still closely related to the previous approaches is *out-of-distribution detection*. Which is concerned with figuring out whether an observations belongs to the same distribution as the training data (Hendrycks and Gimpel, 2017). Unlike the authors describing the previously mentioned approaches, Hendrycks and Gimpel (2017) make no claim about the similarities and differences between *out-of-Distribution detection* and *outlier detection*, *novelty detection* nor *anomaly detection*.

Classification

A classifier commonly refers to a method which assigns observations to a set of predefined classes (the *closed set*). Usually a classifier always assigns a class no matter what. A classifier that can abstain from making such a decision is still not the standard, even though research into this aspect of classification dates back to the origin of pattern recognition (Chow, 1957; Hellman, 1970). These ideas are known nowadays under multiple names such as *classification with reject option* (Herbei and Wegkamp, 2006; Bartlett and Wegkamp, 2008), *learning with rejection* (Cortes et al., 2016), *selective classification* (El-Yaniv and Wiener, 2010) or *open-set recognition* (Majewski and Basztura, 1984; Scheirer et al., 2013).

While the problem of detecting *outliers/anomalies* or *novelties* can be formulated in terms of classification, robust classification system also needs some component to detect *outliers*. Moya and Hush (1996) introduced the term *one-class classifier* to refer to a classifier which generalizes from the classes seen during training, but also correctly rejects any observations which do not belong to those seen during training. *One-class classification* is thus formulated as a measure of generalization capability.

2.1.2 Methods

The many methods to perform open-set recognition can be categorized into three different approaches (Hendrickx et al., 2021). A "detector" component could be put in front of a classifier, to prevent observations that do not belong to any of the known classes to ever reach the classifier.[1] Alternatively, a "detector" could use the output of the classifier to decide whether the decision should be overturned. Or the two components could be integrated into one system, where the classifier itself makes both decisions simultaneously. An example for an "integrated detector"

[1] The problem of designing such a detector is precisely that of *outlier detection*, *novelty detection*, *anomaly detection* and *out-of-Distribution detection*.

that uses the classifier output and an intermediate model representation is *Open Max* (Bendale and Boult, 2016). In this system, a standard deep network classifier is trained on positive samples only. Then a separate statistical model is constructed based on the deep features from the deep network.

Chapter 3

Background

In this chapter, the necessary background information is provided upon which we build our RBF layer later on and evaluate the resulting model.

3.1 Radial Basis Function Neural Networks

A Radial Basis Function is a function $\phi : \mathbb{R}^n \to \mathbb{R}$ which is symmetric around some center $\mu \in \mathbb{R}^n$, as in $\phi(x) = \phi(\|x - \mu\|)$, where $\|\cdot\|$ is a vector norm. A weighted sum of Radial basis functions can be used for approximating or interpolating functions.

$$y(x) = \sum_{i=0}^{m} \lambda_i \phi(\|x - \mu_i\|) \tag{3.1}$$

Broomhead and Lowe (1988) noted that the capability of multi layer perceptron (Rosenblatt, 1958; Steinbuch, 1961) to generalize on unseen data is essentially an interpolation between training samples. At that time, not many of the design decisions involved with creating multi layer perceptron systems were grounded on a strong theoretical basis. As a remedy, Broomhead and Lowe (1988) to use Radial Basis Functions because they are well known function approximators with well studied methods for fitting on data. Thus they suggest a new class of neural network models using Radial Basis Functions. Their proposed architecture is a two layer feed-forward neural network, containing a hidden layer with radial basis functions and an output layer. In the original formulation only the the weights λ_i and the centers y_i in the hidden layer are learned. Such a radial basis function network can solve the exclusive-OR (XOR) problem, which cannot be solved by single layer linear perceptrons (Minsky and Papert, 1969). In fact it has been shown that such radial basis function networks are capable of universal approximation under certain conditions (Park and Sandberg, 1991).

3.2 Deep Radial Basis Function Neural Networks

Deep RBF neural networks were proposed by Lecun et al. (1998) in the variant *LeNet-5* for character recognition. They used a Euclidean RBF as the output layer, with one RBF per class. Thus the most likely class was the one, for which the difference of the previous layer's output and a prototype vector was smallest. The centers of these RBF neurons were not learned, but manually set to some form of prototype drawings of each character. While the CNN architecture has become very popular in deep learning, the RBF variant has mostly been forgotten. Pineda-Arango et al. (2020) suggest that this happened because of the results from Simard et al. (2003), which achieved new

state-of-the art results on the Modified NIST (MNIST) dataset, by using a Softmax output layer trained with Cross-Entropy loss.[1]

Fortunately, deep RBF networks have not been forgotten completely and interest in their study seems to be growing. Zadeh et al. (2018) have recently picked up the study of deep RBF networks, by investigating their robustness to adversarial attacks. They used an RBF kernel with L_1 norm as distance function and showed increased resistance to targeted and untargeted adversarial attacks on the MNIST dataset. Pineda-Arango et al. (2020) studied the sample efficiency of deep RBF networks and found that they achieve higher accuracy when trained with only a small subset of the whole training data on CIFAR-10 & CIFAR-100. They also investigated pretraining a Convolutional Neural Network (CNN) and initializing the RBF centers with k-means clustering. Zhang et al. (2020) studied the performance of adding RBFs before a scaled Softmax layer.

3.3 Open Set Recognition

Open set recognition is a type of *classification with rejection* (Hellman, 1970), where the term *open set* puts the focus on the type of observations that need to be rejected. Specifically, observations that are semantically closely related to those already considered in the training set, because they have been omitted in the sampling of training data in the first place. As an example Majewski and Basztura (1984) performed speaker recognition, but for obvious reasons, the training dataset cannot feasibly contain voices from all people. The set of voices that are not in the training set, are referred to as the *open set*. Ideally, a recognition systems would not only classify those classes correctly that were seen during training, but also correctly abstain from classifying those that were unknown at training time. This property of a recognition system is technically called *generalization*, although it is quite common to use the term *generalization* to refer to the situation where the performance of some measure on the training is highly correlated with the performance on a test set. For practical reasons, the test set is usually an initially fixed subset of the original dataset that was collected.[2] In the laboratory the performance measure of choice is usually the classification accuracy, thus when the accuracy of some method on the test set is good enough, it is usually declared to *generalize well*. However, this type of *generalization* is just one aspect. Outside of the laboratory, researcher were concerned with accuracy in the *open set* setting. They have realized early on, that just classifying those observation in the training set does not translate well into real life performance. Moya and Hush (1996) has used the terminology *within-class*, *between-class* and *out-of-class* generalization to distinguish between the different aspects of it. Particularly, *out-of-class generalization* is the one that is the measure one is interested in for *open set recognition*. As it is the ability to distinguish between the classes of the observations seen during training and any other observation.

From a computer vision perspective, Scheirer et al. (2013) studied *open set recognition* in detail, in particular when the number of unknown classes is very large, such as it is the case in general image recognition. They offer a quantification of the "openness" of a particular recognition problem, in terms of the number of classes used during training and those estimated to be encountered in the wild. They argue that extrapolating far away from the known labeled data, which they call "open space", increases th uncertainty in the classification. They propose the notion of *open space risk*, which is the volume in the representation space assigned to a particular class, but void of samples. To improve the performance of a statistical open-set recognition system, which is usually trained by the minimization of an empirical risk estimate. They argue that, additionally the

[1] Interestingly, they did not only achieve half the error rate compared to Lecun et al. (1998) with their CNN variant, but also more than half the error rate for the MLP baseline trained with Cross-Entropy. They explain this difference with faster computers which allowed them to train much longer.

[2] The history of the MNIST dataset is exemplary of this and will be discussed later on

open space risk should be minimized.

In other words, the recognition function should be "sharp" around the known samples, which has been previously pointed out to be of importance for the case of recognition without complete knowledge of all classes (Bonner, 1966).

More formally, Scheirer et al. (2013) consider a Lebesgue measurable decision function $f(x) = 1$ if x belongs to a particular class k, where $f(x) = 0$ means x does not belong to the class. Let $\mathcal{O}$ be the space assigned to any class and S_O is a large ball which contains $\mathcal{O}$ and all known samples belonging the the class k. Then *open space risk* is defined as the fraction

$$R_{\mathcal{O}}(f) = \frac{\int_{\mathcal{O}} f(x)dx}{\int_{S_O} f(x)dx} \tag{3.2}$$

This formulation of *open space risk* is not necessarily tractable to compute for any f, thus it needs to be approximated in some way (Scheirer et al., 2013). They propose a regularizer for a classifier of the type support vector machine (Cortes and Vapnik, 1995).

3.3.1 Issues in OSR

Dhamija (2022) has summarized current methods for open set recognition and their limitations, like using self-supervised learning, changing network architectures, different loss functions, building probabilistic models based on the deep feature distributions and using additional extra data. A first step is to change the standard Softmax Cross-Entropy loss which assumes a closed world, into a variant that allows for rejection. There are many ways to implement this and even more regularizers. A common approach is to penalize the magnitude of the logits of negative samples as proposed in Dhamija et al. (2018), however the success of this approach is highly dependent on the choice of negative samples. Thus it seems that changes on the loss function are to be necessary but not sufficient. Another popular approach is to use the distribution of deep features to build a statistical model to identify unknown samples. OpenMax proposes to do so using Extreme Value Theory (Bendale and Boult, 2016) to reject samples which are unlike those seen in training. Although this approach sounds good in theory, it does not outperform other OSR methods (Roady et al., 2020; Dhamija, 2022). Walkowiak et al. (2021) gives a potential explanation for this observation: that the inter-class distances often do not follow a Weibull distribution. Recent evaluation protocols for OSR have identified a potential cause for the difficulty in previous approaches, which is that the more similar an unknown class is to those seen during training, the harder it is to discern among them (Palechor et al., 2023).

In our opinion the fundamental issue shared among most approaches is that the deep feature representation of a vanilla CNN does not have the correct metric. As in the distance learned in deep feature space is not equivalent to the natural distance of the objects in the real world. This could be caused by the tendency of large deep neural networks to take *shortcuts* during their training (Geirhos et al., 2020). Which means that network builds a deep representation using the information from features which are merely correlated within a class in the dataset, but not actually predictive of the correct class. For example in image recognition of animals, the performance drops significantly when the location of the camera is moved (Beery et al., 2018). This is a more widespread problem in image recognition, as it was observed that deep networks mostly rely on the background of the objects (Xiao et al., 2020).

3.4 Deep learning in image recognition

The idea of artificial neural networks has been studied under various terms and also rebranded numerous times in the past. Nowadays it is very well known under the term *deep learning* (Good-

fellow et al., 2016). The term itself has been introduced originally by Dechter (1986) in a adjacent field. In general, *deep learning* refers to the stacking of multiple *layers* containing "artificial neurons", which process their respective input and pass along their output. In classic *feed-forward networks*, input and output are normally modeled as a vector with elements from $\mathbb{R}$. The idea of stacking layers of artificial neural networks was already present in the works of Rosenblatt (1958) and Steinbuch (1961). The next step was to find methods to train such networks efficiently. Initially, such networks were trained layer by layer (Ivakhnenko, 1971). Then it was suggested to train the whole network *end-to-end* by using *stochastic gradient descent* methods (Amari, 1967; Robbins and Monro, 1951). Linnainmaa (1970) described the method of *backpropagation*, which is an efficient way to optimize arbitrary neural network topologies. Training such deep neural networks became only practical recently, due to their immense computational costs. Cireşan et al. (2010) showed that utilizing powerful parallel computing devices like GPUs, the *end-to-end* training of such deep networks is not only feasible, but they also exhibit very high performance on contemporary benchmarks. A modern deep learning system can be organized into three subsystem: *feature extraction*, *deep features representation* and *classification*.

3.4.1 Feature extraction

A *feature* is a variable used in a statistical model that hopefully contains some information useful for the modeling task. Traditionally, features were carefully engineered by hand in a preprocessing step, such that they allow for linear separation (Nagy, 1968). However, the motto of deep neural networks is to learn everything *end-to-end*, therefore also the feature extraction should be done in an automatic fashion. Unfortunately this task is very difficult for image recognition for two intertwined reasons. First, image data is usually very high dimensional, especially if they are colored. Thus naively processing an image in a neural network layer wise requires a number of artificial neurons equal to the number of pixels times color channels. Stacking such layers of similar size becomes quickly very computationally expensive. The second related problem is that such a large number of artificial neurons is not only expensive in terms of compute, but also in terms of samples. Because a large number of neurons means that a large number of parameters need to be estimated from data, thus a lot of data is needed. The seminal works of Fukushima (1980) introduced an architecture called *neocognitron*, which is nowadays much better known under the term CNN. The main idea is to use an efficient sharing of parameters by applying the *convolution* operation on a 2D representation of an image and a *kernel* of much lower dimensionality. The result is then reduced by a so called *pooling* operation, which locally either combines several outputs or selects one *e. g.* the largest, which is called *max pooling*. Additionally, a nonlinear activation function is then applied (Goodfellow et al., 2016, p. 326 ff). Two popular CNNs based neural network architectures are *LeNet* and *ResNet*.

LeNet variant

LeNet networks were originally introduced by Lecun et al. (1998). Wen et al. (2016) proposed a modification termed *LeNet++* which is deeper and wider, but compresses the deep features in the fully connected layer to only two dimensions. This has the advantage that the deep feature representation can be directly visually inspected, with acceptable loss of learning capacity. This variant was adapted by Dhamija et al. (2018), where additionally batch normalization is used after each convolution block (Ioffe and Szegedy, 2015).

ResNet-20

For more difficult image recognition problems, such as recognizing cars or animals, a different network is needed. While the class of functions required to separate digits in a black and white representation is very complex, the topology of these forms is still somewhat simple. For example the MNIST digits consist of lines and at most two loops. The topology of cars and animals is much more complex and additionally involves information about colors and local texture. Theoretically, it is well known that very complex functions can be approximated more efficiently in certain layered compositions. In terms of neural networks this means that certain deeper neural networks can represent very complex functions much more efficiently than shallow but wide neural networks (Pascanu et al., 2014; Montufar et al., 2014). However, while simply stacking many layers upon layers creates a theoretically powerful deep neural network, it is not feasible to train it with back propagation, due to the well known vanishing gradient problem (Hochreiter, 1998). For general recurrent neural networks (RNN) this problem can be circumvented with a complex neural network architecture like the *LSTM* (Hochreiter and Schmidhuber, 1997). Srivastava et al. (2015a) applied the same idea to convolutional neural networks (CNN) in an architecture termed *highway networks*. It builds upon the idea of using so called *gates* to control the flow of information throughout the network. In a *highway networks* information can "skip around" layers with non-linear activation functions unmodified, thus on a backward pass, on these *highways* the gradients don't vanish. If we denote the output $\mathbf{y}$ of a classic feedforward neural network layer with

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W_H}) \tag{3.3}$$

where H is the non-linear transform using the weights $\mathbf{W_H}$, then in general a *highway network* has the form

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W_H}) \cdot T(\mathbf{x}, \mathbf{W_T}) + \mathbf{x} \cdot C(\mathbf{x}, \mathbf{W_C}). \tag{3.4}$$

T is called a transform gate and controls how much weight should be given to the transformation H. This has to be balanced with the *skipping* of information, which is controlled by the carry gate C. This class of deep neural networks allows the training of very deep networks with over 100 layers (Srivastava et al., 2015b) using SGD. A special type of this network class is obtained if both gates are open and constant, meaning the transform gate and the carry gate are both fixed to 1.

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W_H}) + \mathbf{x}. \tag{3.5}$$

Networks of this type are better known under the term *ResNets* (He et al., 2015a). *ResNets* have gained widespread popularity after such an network with 152 layers won the ImageNet 2015 competition (Russakovsky et al., 2015).

3.4.2 Deep Features Representation

For a long time it was computationally infeasible to extract such features directly from data end-to-end as discussed in detail in Section 5.2. With CNN based architecture the input signal can be *compressed* to some reasonable dimensionality, the result of this projection is nowadays referred to as *deep features*, because they are taken from the deeper layers in a deep neural network. The precise nature and meaning of these *deep features* are not well understood. This is not at all unexpected, since a complete account of *deep features* would provide all necessary explanations on the "magic" of at least *some* types of neural networks.

Nonetheless, there are a few interesting hypotheses about these *deep features*. In the case of a CNN: one approach views these *deep features* as an abstract high-level representation of the input. Where the deepest features are the most abstract in a hierarchy of concepts. The reasoning

goes as follows: since the first layer detects edges and their orientation and the second layer detects a combination of such edges thus the higher level detect parts of objects and finally the specific object in question (LeCun et al., 2015). In that view, the *deep features* should be very sparse, having only high activity for the specific concept detected and no activity for the rest. Notice that this view is essentially an argument by analogy, since CNN were originally constructed by Fukushima (1980) based on insights from Hubel and Wiesel (1962) with their research of the visual cortex. Therefore, image recognition in animals probably works like a CNN, thus the deepest layers represent very high-level concepts, *e. g.* a specific person (Goodfellow et al., 2016, p. 358ff).

However, for very deep networks the situation changes a bit. Such networks usually employ *skip-connections* and work very well for all sorts of tasks. But by using *skip-connections*, the input is no longer purely abstracted in a sequential hierarchy, since the input signal is somewhat "passed-through" at every supposedly *level of abstraction*. Greff et al. (2017) provides an explanation in that not ever convolution layer creates a new abstraction level, but that a new level possibly arises at the transition to a different dimensionality in the representation. This complicates the picture quite a bit of what the output of a bunch of stacks of convolution layers actually represents. From this perspective it is not immediately obvious what the *deep features* exactly are and how many of those are needed for a particular task, respective how to choose the dimensionality. However, we can look at the CNN as a non-linear dimensionality reduction method, which is subject to study in the field of *manifold learning* (Cayton, 2005). Which studies the question of how to represent a high-dimensional input signal faithfully in lower dimensions and works with the assumption that the signal in most high-dimensional data that we usually are interested in, lie in fact on a manifold of much lower dimensionality. This assumption is nowadays also known under the term *manifold hypothesis* (Narayanan and Mitter, 2010). The dimensionality of this manifold would provide a potentially useful reference point to pick the dimensionality of the *deep feature space*. This is also called the *intrinsic dimension* of the data. In theory there exists a plethora of methods to estimate it, but the practical application in high dimensions is far from trivial (Grassberger and Procaccia, 1983). Estimates from different methods do agree somewhat for an *intrinsic dimension* of the digits in the MNIST dataset in the range of 8-15 (Hein and Audibert, 2005; Costa and Hero, 2006). Whereas the *intrinsic dimension* of the Canadian Institute For Advanced Research (CIFAR) datasets is possibly twice of that (Pope et al., 2021). Note that there are certain assumptions on the additive noise component of the data that makes it high-dimensional in the first place. Depending on the magnitude of the noise compared to the signal, a different approach might be needed (Blanchard et al., 2006). However, the further study of this matter and the relation of the *intrinsic dimension* to the dimension of the *deep features* is out of scope of this thesis, thus we defer further investigations to later works (Pestov, 2007; Fefferman et al., 2016; Ansuini et al., 2019).

Another interesting characterization of the *deep feature representation* is through the lens of coding theory. The idea of *disentangled representation* is to find features that are independent of the others (Schmidhuber, 1992), which is based on ideas to remove redundant signals in the representation (Barlow, 1989). Such *disentangled representations* are rather useful: *e. g.* to disentangle between shapes and styles of MNIST letters (Chen et al., 2016).

3.4.3 Classification

In a deep learning based pattern recognition system, the last layer is usually used to perform the classification step. If this model is designed with the knowledge about K classes then the last layer can be constructed with K elements. As a decision rule the index of the largest element can be used to assign a class, *i. e.* one applies the $\arg\max$ function on the last layer.

Since the $\arg\max$ function is not differentiable another method is needed. The differentiable *Softmax* function, as introduced by Bridle (1990) approximates the $\arg\max$ function and can thus be used with gradient descent learning methods. Let a model $f : \mathbb{R}^{\text{input}} \to \mathbb{R}^K$ map the input x

to K values z_k. Then the *Softmax* function can be used to approximate $\arg\max$ by squashing each *logit* z_k for $k \in \{1, .., K\}$ to the domain $[0, 1]$:

$$\text{softmax}(z)_k = \frac{e^{z_k}}{\sum_j^K e^{z_j}} \tag{3.6}$$

The *Softmax* function therefore can be used to estimate the probability of a sample x belonging to class k:

$$\hat{p}_k(x) = \text{softmax}\left(f\left(x\right)\right)_k \approx p(k|x) \tag{3.7}$$

Note that the normalization $\sum_j^K e^{z_j}$ assumes that exactly K classes are possible, thus all inputs are mapped to one those. If more states than the known K are possible, the output of $\text{softmax}(z)_i$ becomes a biased estimate of the true class probability $p_k(x)$.

The standard loss function for neural networks with a *Softmax* classification layer is the *Categorical Cross-Entropy* loss:

$$\mathcal{J}_{CCE}(x) = -log\left(\hat{p}_k\left(x\right)\right) \tag{3.8}$$

Entropic Open-Set loss

When adding negative samples $x \in \mathcal{D}_n$ during training the standard *Categorical Cross-Entropy* loss can not directly be used, because Equation 3.8 is 0 for all negative samples. One way to overcome this issue is by adding a special extra *background* class such that the problem is translated into a different classification among $K + 1$ classes. One problem with this approach is that it assumes that all the possible negative samples forming the *background* class are similar, which is a very unlikely assumption. Alternatively, one can enforce an *uncertain* output of the classifier for negative samples. Dhamija et al. (2018) introduced an objective function to train statistical multi-class models that utilize the *Softmax* function, such that the resulting probabilities more faithfully reflect the uncertainty of the model for a particular decision. This is achieved by modifying the standard categorical cross-entropy loss to enforce a maximum entropy distribution for observations that are known to not belong to any of the classes of interest.

$$\mathcal{J}_{EOS}(x) = \begin{cases} -\ln\left(\hat{p}_k\left(x\right)\right) & \text{if } x \in \mathcal{D}_p \text{ belongs to class } k \\ -\frac{1}{K}\sum_{k=1}^{K}\ln\left(\hat{p}_k(x)\right) & \text{otherwise} \end{cases} \tag{3.9}$$

Because of the normalization in the *Softmax* function Equation 3.6, for negative samples, the loss $\mathcal{J}_{EOS}(x)$ is minimal when all $\hat{p}_k$ are equal, thus when $\hat{p}_k(x) = \frac{1}{K}$. In terms of the *logits* z_k, there are multiple local minima, namely whenever all $z_i = z_j, \quad \forall i, j \in \{\, 1, .., K \mid i \neq j \,\}$.

Dhamija et al. (2018) introduced another loss called *Objectosphere loss*, which attempts to overcome the multiple minima issue with *Entropic Open-Set loss*. By adding an additional term that penalizes large deep feature activations for negative samples and enforces a minimal deep feature activation for positive samples. However, this loss adds two additional hyperparameters that need careful tuning through cross-validation. For the sake of scope of the thesis, this loss will no be considered in the experiments. While in the experiments in from Dhamija et al. (2018) show that *Objectosphere loss* performs better than *Entropic Open-Set loss* on some datasets, *Entropic Open-Set loss* in turn performs better than a *background class* approach. The improved performance for *Entropic Open-Set loss* has also been observed on large dataset like *ImageNet* (Palechor et al., 2023).

3.5 Exemplar and Prototyp-based Classification

Classifying objects into groups is a very natural activity for humans and essential for any image recognition in living beings. The mechanisms of how this is performed in humans is studied within the framework of the *cognitive sciences*. In that literature this mechanisms is also known under the term *categorization* (Seger and Miller, 2010). There are many theories about *categorization*, but most of them can be grouped into two approaches. In brief terms, theory of *Prototypes* assumes that classification is performed by converting an observations into an abstract higher level representation which contains all the defining features of a particular class. Then this abstract representation is compared to those in memory to find the appropriate *prototype* which defines the class. Alternatively, the theory of *exemplars* directly compares the observation with those in memory to find the one that is most similar, which in turn defines the class Jäkel et al. (2008). Note the similarities of the *exemplar*-based classification with the well known nearest neighbor classification method (Hellman, 1970).

Chapter 4

Data

The scope of this work is restricted to the task of classification in the image domain. However even after this restriction, there are still dozens of potential datasets publicly available for experimentation. This chapter will describe the used dataset in detail, along with the justification for their usage in this work.

4.1 Closed-set

We refer to the original source datasets as *closed-set datasets* to emphasize that usually the task performed on these is *closed-set image recognition*. In Table 4.1 a list of the used datasets is shown with their respective sizes.

4.1.1 MNIST

The most popular dataset for the task of image classification is undoubtedly MNIST (Bottou et al., 1994; LeCun et al., 1995). Its small data size per sample (28×28 gray scale pixels) and low number of classes (10), makes the MNIST dataset ideal for development and verification purposes of the RBF layers, because it requires comparatively less computing resources for training and allows for informative visualizations of its ten classes. The MNIST dataset contains 60k training and 10k testing images. The MNIST dataset is based on the National Institute of Standards and Technology (NIST) special database 19 (Grother and Hanaoka, 1995), containing handwritten digits and letters. A competition on the NIST dataset distressed some researchers, because the training data for the competition was created by paid workers, whereas the test data was collected from high school students (LeCun et al., 1995). They argue that in this case, the training and test data belong to different distributions, thus preventing the use of many machine learning methods. Subsequently, they modified the NIST dataset by creating a new training/test split, where the digits of paid workers and unpaid students are mixed.

While this dataset is useful for development purposes, its value for benchmarking and comparing different methods is nowadays very limited. The original *LeNet* CNN variant achieves an error rate of $0.95\% - 1.1\%$ (Lecun et al., 1998). Modern deep learning architectures achieve an error rate in the range of 0.23% (Cireşan et al., 2012), which translates to 23 mistakes on the testset. Such a high classification accuracy raises questions about the error rate of the labels the dataset itself (Cohen et al., 2017). The original estimate of human performance was 0.2% (LeCun et al., 1995). An extensive analysis of the MNIST dataset estimates the label error rate in the test set as 0.15% (Northcutt et al., 2021). Thus it can be argued that the MNIST dataset is solved with CNNs.

Table 4.1: CLOSED-SET DATASET. *Source datasets used to construct the open-set dataset variants*

			Total Size	
	Resolution	Classes	Training	Test
MNIST	$28 \times 28 \times 1$	10	$60k$	$10k$
EMNIST-letters	$28 \times 28 \times 1$	26	$124k$	$20k$
KMNIST	$28 \times 28 \times 1$	10	$60k$	$10k$
CIFAR-10	$32 \times 32 \times 3$	10	$50k$	$10k$
CIFAR-100	$32 \times 32 \times 3$	100	$50k$	$10k$

To study differences in CNN based classifiers a different dataset should be used, fortunately the selection of images is just a subset of the much larger NIST dataset.

4.1.2 EMNIST

The original NIST dataset contains images with 128×128 pixels (Grother and Hanaoka, 1995) and provides not just images of digits, but also lower and uppercase letters. However, it requires some more preprocessing compared directly using MNIST, which could be a contributing factor for explaining the popularity of the derivative dataset MNIST. To improve the accessibility of the original dataset, Cohen et al. (2017) introduced the Extended Modified NIST (EMNIST) dataset. It follows the same preprocessing steps and provides 28×28 pixel images. It is also much larger than MNIST, by providing 280k images for the task of digit recognition.

The EMNIST dataset consists of multiple subdatasets: *digits* with 10 classes, *letters* with 26 classes (lower and uppercase), and a balanced dataset containing digits and letters, where certain lowercase letters are merged with their uppercase version[1] with a total of 47 classes. Compared to MNIST, EMNIST is a substantially harder classification problem. The baseline provided in (Cohen et al., 2017) achieves an accuracy of 95.9% for the digits subset and 78.0% for the balanced subset. This baseline is a form of multi layered perceptron with random weight initialization, where only the final readout layer is trained (Rosenblatt, 1962).

4.1.3 KMNIST

For quite some time, the MNIST dataset has been the measure used to compare different algorithms and methods. While the benchmarks nowadays use different dataset, the "*-MNIST" format itself has spread to other datasets. One of them is Kuzushiji-MNIST (KMNIST), which is a dataset with the same format as MNIST, but with different content (Clanuwat et al., 2018). *Kuzushiji* refers to an old cursive Japanese script, which is no longer taught nowadays and thus can only be read by certain scholars. The creation of this dataset has been partially motivated by the approximately two million books, written in *Kuzushiji*, that cannot be read anymore by the general public. The dataset contains $60k$ training and $10k$ testing images of size 28×28 pixels divided into 10 classes, just like MNIST. KMNIST is slightly more difficult to learn for machines, possibly due to an increased intra-class variance. *Kuzushiji* consist of 49 character, out of those the authors have selected 10. However, in classical Japanese, some characters can be written in multiple distinct forms. For an example, see the *Hiragana* "tsu" in Figure 4.1. This property of the dataset might be of interest for openset recognition tasks.

[1] the letters c,i,j,k,l,m,o,p,s,u,v,w,x,y,z

Hiragana	Unicode	Samples	Sample Images
お (o)	U+304A	7000	
き (ki)	U+304D	7000	
す (su)	U+3059	7000	
つ (tsu)	U+3064	7000	
な (na)	U+306A	7000	

Figure 4.1: KMNIST EXAMPLE HIRAGANAS. *"tsu" has multiple forms for the same character. Source: Clanuwat et al. (2018)*

4.1.4 CIFAR-10/100

While handwriting recognition is already a difficult task for machines to perform, the visual complexity in the real world is undoubtedly much larger. Handwriting systems are still limited to a couple dozens of classes with certain variations, but there exist innumerable visual *classes* in the natural world. The further study and subsequent teaching of machines to understand natural visual scenes requires data, quite a lot of it. The proliferation of the Internet and its associated growth of search engines enabled the economical creation of such datasets. Torralba et al. (2008) collected millions of images by entering English nouns into search engines and downloading the resulting images. The list of nouns in turn was readily made available by the *WordNet* database (Miller, 1995). One issue with image recognition is that the data is very high dimensional, usually in terms of pixels: width $\times$ height $\times$ channels. Which leads to two issues, first already the storage of the dataset take up huge amount of space and second is machine learning in high dimensions very expensive in terms of compute. Thus using the smallest possible image that still allows for recognition would be appropriate. Torralba et al. (2008) experimentally measured the recognition rate of humans on different datasets as a function of image resolution and discovered that even with a resolution of 32×32 pixels (color image), human could still correctly recognize about $85\% - 90\%$ of scenes/objects. Therefore their dataset chose this image dimension as a trade-off. One issue with this dataset is that the labels are noisy, because the label is the term used to search for it. This does not necessarily mean that an image returned by the search engine actually contains the search term in the image. The get better labels for the images, Krizhevsky (2009) coordinated a manual labeling of a subset of these images. Human raters selected thus those images that are clearly identifiable as the label and are photo-realistic. They created two datasets by selecting 10 classes with $6k$ images each and another one consisting of 100 classes with 600 images. Both received a train/test split of $5:1$. The resulting datasets are now widely known as *CIFAR-10* and *CIFAR-100*. Where the name refers to the funding institute CIFAR. The classes of *CIFAR-10/100* are mutually exclusive, where *CIFAR-100* was explicitly created to provide a set of negative samples for the *CIFAR-10* image recognition task.

4.2 Open set

In this work an *open-set dataset*, denotes a set of sets containing a regular dataset for the positive samples and another for the negative samples. Evaluation might be done on a third dataset or on parts of the negative dataset. The naming of these derived datasets is constructed by the origin of their respective *positives-negatives-unknowns* source dataset. These subsets are abbreviated with $\mathcal{D}_p$ for the *positives*, $\mathcal{D}_n$ for the *negatives* and $\mathcal{D}_u$ for the *unknowns*. All the source dataset were obtained from the *PyTorch* package. The used source datasets $\tilde{\mathcal{D}}_x$ consist of some classes $\mathcal{C}_x$. In general the classes of some datasets $x, y, ..$ can be split into two disjoint sets: a "known" set $\mathcal{C}_p$ and a proxy for all other possible inputs, the "unknown" set $\mathcal{C}_u$. In the used setup here, two source datasets x, y are split such that $\mathcal{C}_p \subseteq \mathcal{C}_x, \mathcal{C}_n \subset \mathcal{C}_y$ and $\mathcal{C}_u \subset \mathcal{C}_y$ with the constraint that $\mathcal{C}_n \cap \mathcal{C}_u = \emptyset$. In other words one dataset is used as-is and a second dataset is split class-wise into a negative sample set used during training and an unknown test set used to evaluate the *open-set recognition performance*. This way data from $\mathcal{D}_u$ was never seen during training, but similar data $\mathcal{D}_n$ which was collected in the same manner, but depicts different information.

The used source datasets $\tilde{\mathcal{D}}$ provide a standard training/test split $\tilde{\mathcal{D}}^{\text{train}} \subset \tilde{\mathcal{D}}, \tilde{\mathcal{D}}^{\text{test}} \subset \tilde{\mathcal{D}}$ such that $\tilde{\mathcal{D}}^{\text{train}} \cap \tilde{\mathcal{D}}^{\text{test}} = \emptyset$. We keep the train/test split for $\mathcal{D}_p$, whereas $\mathcal{D}_n$ is constructed only from the train split and conversely $\mathcal{D}_u$ only contains the test split from the source dataset. Thus the *open-set datasets* can be characterized by the following four cardinalities $|\mathcal{D}_p^{train}|, |\mathcal{D}_n^{train}|, |\mathcal{D}_p^{test}|$ and $|\mathcal{D}_u^{test}|$.

4.2.1 Digits-Letters-Letters

This *open-set dataset* was constructed by taking all classes from MNIST as $\mathcal{D}_p$ and normalize the data with $\mu = 0.1306, \sigma = 0.3081$. The negative set $\mathcal{D}_n$ consists of the first 8 classes from the EMNIST dataset and the subsequent 18 are used as unknowns $\mathcal{D}_u$. Samples from this combined *open-set dataset* are shown in Figure 4.2. The resulting dataset cardinalities are $|\mathcal{D}_p^{train}| = 60000, |\mathcal{D}_n^{train}| = 38400, |\mathcal{D}_p^{test}| = 10000$, and $|\mathcal{D}_u^{test}| = 10000$. This dataset is unbalanced in terms of the number of classes used as unknowns and negatives, $|\mathcal{C}_u| > |\mathcal{C}_n|$. Due to the number of available samples in $\mathcal{D}_n$ the resulting subsets used during training are also unbalanced. In preliminary experiments a balanced split has been used, but the differences in the used methods was rather small due to saturation of the performance measure. Thus with this unbalanced split the task is made slightly more difficult for a network to learn.

4.2.2 Kuzushiji-Letters-Letters

This dataset was constructed analogously to the previous one (*Digits-Letters-Letters*) but uses KMNIST instead of MNIST for $\mathcal{D}_p$. Because *closed-set* accuracy on KMNIST is slightly lower than on MNIST, meaning that the task is more difficult, no unbalanced class split has been performed. Thus the negative set $\mathcal{D}_n$ consists of the first 13 classes of EMNIST and the unknowns $\mathcal{D}_u$ of the rest. The resulting dataset cardinalities are therefore $|\mathcal{D}_p^{train}| = 60000, |\mathcal{D}_n^{train}| = 60000, |\mathcal{D}_p^{test}| = 10000$, and $|\mathcal{D}_u^{test}| = 10000$. Examples from the dataset are shown in Figure 4.3.

4.2.3 CIFAR10-50-50

The *open-set dataset* for natural image recognition is based on the CIFAR datasets. *CIFAR-10* is used as $\mathcal{D}_p$. The first 50 classes from *CIFAR-100* as $\mathcal{D}_n$ and the rest for $\mathcal{D}_u$. The resulting dataset cardinalities are therefore $|\mathcal{D}_p^{train}| = 50000, |\mathcal{D}_n^{train}| = 25000, |\mathcal{D}_p^{test}| = 10000$, and $|\mathcal{D}_u^{test}| = 5000$.

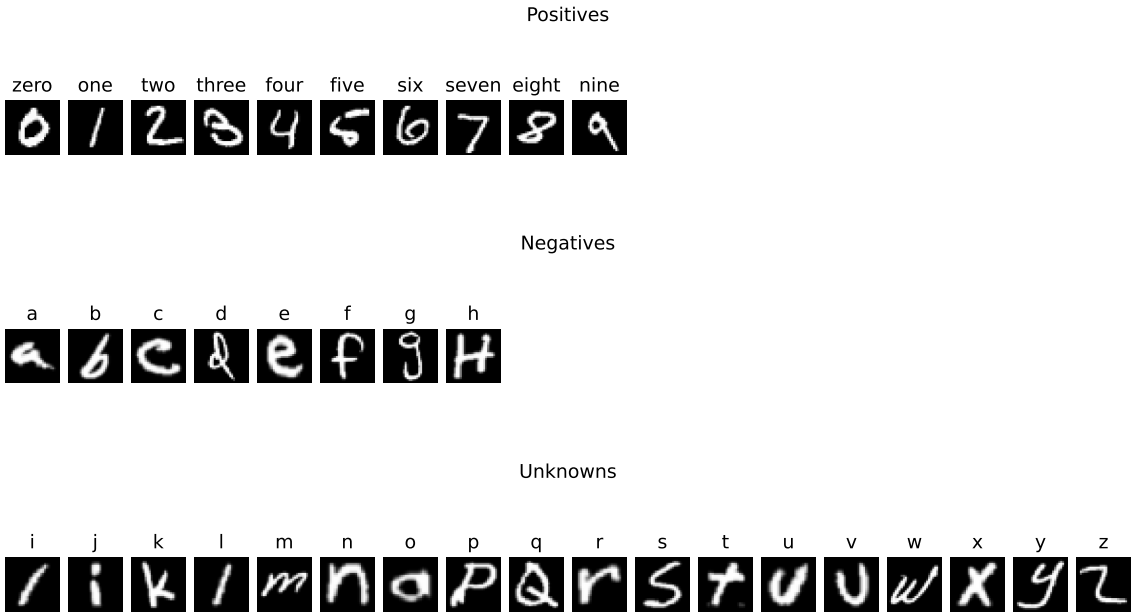


Figure 4.2: SAMPLE IMAGES FROM DIGITS-LETTERS-LETTERS DATASET.

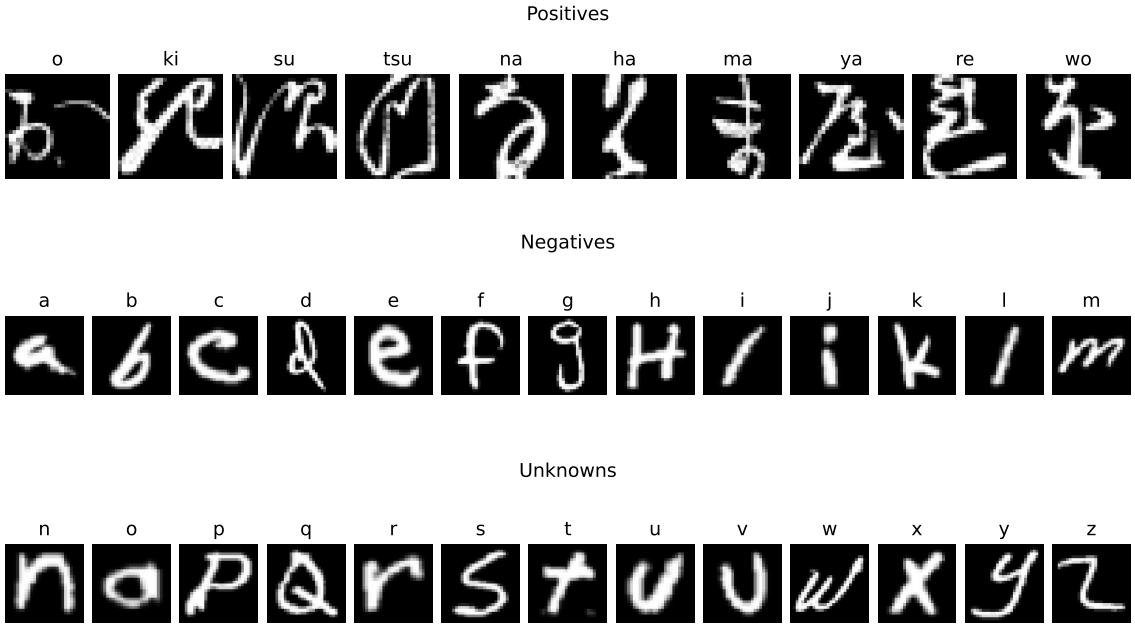


Figure 4.3: SAMPLE IMAGES FROM KUZUSHIJI-LETTERS-LETTERS DATASET.

Figure 4.4: SAMPLE IMAGES FROM CIFAR10-50-50.

Chapter 5

Problem Analysis

In this chapter we discuss the use of radial basis functions in deep neural networks and the confounding issues in *open-set recognition*.

5.1 Deep RBF network design space

As per our first research question RQ-1, we are interested in the different ways that *radial basis function* can be parametrized in general and utilized in a deep neural network.

We will call a deep neural network which uses *radial basis function units* in any layer a *deep RBF network*. A *RBF unit* is a type of artificial neuron, which responds proportional to the similarity of the input to some other point μ. We will refer to μ as the *center* of the unit. The similarity is computed using a distance function $\delta : \mathbb{R}^d \times \mathbb{R}^d \to [0, \infty)$ and a center μ. An activation function $\psi : [0, \infty) \to \mathbb{R}$ modulates the response of the unit. The general form of a *radial basis function* ϕ is therefore

$$\phi(\mathbf{x}) = \psi(\delta(\mathbf{x}, \mu)) \tag{5.1}$$

The shape of many activation functions can be tuned with another parameter σ, which is often called the *width* or *scale* of the basis functions (Ghosh and Nag, 2001). As a general and flexible formulation, we will let ψ, δ, μ and σ be parameters $\theta = \{\psi, \delta, \mu, \sigma\}$ and refer to parametrized *radial basis function* ϕ as the *radial basis function unit*: RBF_θ.

$$RBF_\theta(\mathbf{x}) = \psi(\delta(\mathbf{x}, \mu); \sigma) \tag{5.2}$$

In Table 5.1 a selection of deep RBF models is listed with their specific parametrization. The notation is adapted such that σ denotes the scale parameter of the activation function ψ. A common choice for the activation function ψ is the parametrized *Gaussian function* $\exp(-\alpha x^2)$. As a distance function the *Euclidean* distance is usually used:

$$\delta_{Euc}(\mathbf{x}, \mu) = \sum_{i=1}^{d} (x_i - \mu_i)^2 \tag{5.3}$$

In principle, such *radial basis function units* can be used in any layer of a deep neural network architecture. We can denote the first layer as the one receiving the networks input and the last one producing the output. Most deep RBF networks position the RBF at the last layer L (Lecun et al., 1998; Zadeh et al., 2018; Pineda-Arango et al., 2020; van Amersfoort et al., 2020), sometimes as input to the *Softmax* function (Zhang et al., 2020). A rather rare positioning is in the earlier layers

Table 5.1: DEEP RBF SURVEY. *Some deep RBF models from the literature are categorized in terms of their parameterization and topology. L^{-n} denotes the nth last layer and C denotes the convolutional layers.*

			Parametrization		
Work	Placement	RBF	Hyperparameter	Learnable	Softmax
Lecun et al. (1998)	L	$\|\mathbf{x}-\mu\|_2^2$	(μ)	(μ)	
Tabernik et al. (2016)	C	$\frac{\exp(-\sigma\|\mathbf{x}-\mu\|_2^2)}{\sum_x \exp(-\sigma\|\mathbf{x}-\mu\|_2^2)}$		μ, σ	
Zadeh et al. (2018)	L	$\|W^\intercal \mathbf{x}+b\|_p^p$	p	$\mathbf{W}, b$	
Amirian and Schwenker (2020)	L^{-1}	$1-\frac{(x-\mu)^\intercal D(x-\mu)}{\sigma^2}$		$\mathbf{D}, \mu, \sigma$	✓
Pineda-Arango et al. (2020)	L	$\exp(-\sigma\|\mathbf{x}-\mu\|_2^2)$	σ	μ	
	L	$\frac{1}{\|\mathbf{x}-\mu\|_2^2+\sigma}$	σ	μ	
Zhang et al. (2020)	L	$\exp(-\sigma\|\mathbf{x}-\mu\|_2^2)$	σ	μ	✓
van Amersfoort et al. (2020)	L	$\exp(-\sigma\frac{1}{n}\|\mathbf{W}\mathbf{x}-\mu\|_2^2)$	σ	$\mathbf{W}, \mu$	

in the case of a CNN. Such a scheme has been proposed, where a type of RBF is replacing a filter in the convolutional layers C (Tabernik et al., 2016; Tabernik, 2021),

5.2 The open set recognition problem

In RQ-2 we ask about the confounding factors in the *open-set recognition problem* for deep neural networks. A general description of *open-set recognition* has been presented in Section 3.3.

We are now concerned with the precise definition of the *open set recognition problem*.

In general terms, the *open set recognition problem* is concerned with teaching a machine to recognize certain items in the real world under resources constraints. This constraint means that usually only a finite, potentially rather small set of data is available to teach the machine and that the learning should be quick in terms of time and compute cost. Otherwise the trivial solution would be allowed to collect and memorize all items in the reachable world. The prefix *open set* is a historical artifact, because general pattern recognition is a very hard problem. Understandably, researchers have initially focused on a much simpler but still very difficult problem: *closed set recognition*. Therein, the set of items is artificially constrained to contain only very few items. A lot of progress in pattern recognition happened using methods evaluated on such artificial tiny subsets of all possible items. The problem arises when such systems leave the lab and are deployed in the real world where they are exposed to the full set of naturally occurring items.

In the domain of images, we argue that for example the general *image recognition problem*, in the spirit of a thinking machine, is not constrained to some closed set, but in fact open. Therefore *open set image recognition* is equivalent to the general problem of *pattern recognition* where the patterns are images. Thus we can resort to the vast literature on *pattern recognition* for insights. Recall that in machines *pattern recognition* is fundamentally concerned with the linear separation of some input signal $\mathbf{x}$. In the case of two items of interest a hyperplane $\mathbf{W}$ that separates the two classes is needed (Bishop, 2006, p. 181).

$$y(\mathbf{x}) = \mathbf{W}\mathbf{x} + b \tag{5.4}$$

The theoretical problem of finding such a $\mathbf{W}$ is extensively studied and a multitude of fast solver are known given certain assumption of $\mathbf{x}$ (Nagy, 1968; Bishop, 2006).

The biggest practical problem has been, since the beginnings, actually computing these methods for a large signal $\mathbf{x}$. For illustration purposes, let us assume $\mathbf{x}$ consists of a grayscale im-

age with $w \times h$ pixels. There is a dataset of n images, thus all the data can be represented as $\mathbf{X} \in \mathbb{R}^{n \times (h \times w)}$. A classic method to find $\mathbf{W}$ in Equation 5.4, under certain assumption, is the ordinary least square approximation, which can be computed as follows:

$$\hat{\mathbf{W}} = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{y} \tag{5.5}$$

Let us abbreviate the dimension of $\mathbf{X}$ with $p = w \times h$ then the computational complexity of Equation 5.5 is

$$\mathcal{O}(p^2 n + p^3) \tag{5.6}$$

These number grows rather quickly, some quick napkin math tells us that a reasonable image of side length 1024 pixel requires

$$((10^3)^2)^3 = 1 \times 10^{18}$$

operations, which requires about a day of computing time on a modern GPU[1]. While there are other more sophisticated methods to solve Equation 5.4 with a smaller complexity in terms of input dimension p like SVMs, the complexity is paid instead by the number of samples n (Cortes and Vapnik, 1995). The main problem is therefore that statistical learning on the raw input with a high dimensionality is incredible demanding in terms of computing resources and that without hue amounts of data, the model would overfit to the training data. The solutions that lead to the proliferation of deep learning is to use approximation methods. In general such methods work by projecting the input $\mathbf{x}$ to another space by using certain functions ϕ, wherein the data becomes separable (Nagy, 1968).

$$z(\mathbf{x}) = \sum_i^m w_i \phi_i(\mathbf{x}) \tag{5.7}$$

If m is equal than the dimension of $\mathbf{x}$ this is equivalent to Equation 5.4, however if it is smaller we are effectively projecting to a representation of lower dimensionality. In general this is not possible without any loss of information, but for inputs with certain structure it might be. Let us assume for now that $m \ll d$ without deleterious loss of information. Now for closed-set recognition a discriminative classifier is needed operating on the $z(\mathbf{x})$ representation, for which many methods are well known to run in acceptable time (Bishop, 2006). Many of these methods actually converge to the correct classifier on average, given a very lager dataset to train from (Vapnik, 1991).

A problem of *open set recognition* becomes now visible. So far methods are only known to work well with a very large dataset covering all classes, meaning properly sampling the underlying data distribution. Unfortunately, a defining property of *open set recognition* is that we do not have access to the full data distribution to sample from and train a model with. In general, this situation is hopeless, but for many practical scenarios such as the recognition of natural images there is much more structure in the data that we might be able to exploit. A common assumption in *pattern recognition* is that the high dimensional input data contains a signal of much lower dimensionality, this assumption is sometimes called the *manifold hypothesis* (Cayton, 2005). This means that input data containing a relevant signal will lie on or close to a manifold which is locally Euclidean. This assumptions has some intuitive plausibility, imagine a picture of some object and apply certain transformation like rotation or translation. The resulting pictures will show large differences in the input encoding in high dimensions, but they all result from the same object, which must thus have some lower dimensionality. In the visual domain, empirical evidence indicates that the state space of natural images is lower than the data dimension in pixels (Lee et al., 2003). Thus with the properly chosen non-linear dimension reduction of high-dimensional input data,

[1] assuming 1 TFLOPS using single precision floating point numbers

the standard classifier methods should be still applicable. Unfortunately this task is far from easy (Dhamija, 2022). The current work is precisely motivated by this difficulty. There are at least two ways in the pattern recognition "pipeline" where difficulties arise in *open-set recognition*. One popular explanation attempt is the idea of "open space risk". Scheirer et al. (2013) argues that in most recognition models the *decision region* is effectively unbounded and therefore even samples belonging to none of the interested classes always get classified. This idea proposes that *decision region* spanned by the decision hyperplanes of a model should be finite and very small (Scheirer et al., 2014). While this idea was originally developed on classifiers that use separately extracted features, it was also extended to deep networks, where we have again the case as described above (Bendale and Boult, 2016). We call this idea the *open-space risk hypothesis*.

To explain the second hypothesis, we will first set up the necessary formalism to allow for better distinction between the two.

Let us thus assume that the set of natural objects $\mathcal{O}$ belonging to C classes, these object reside in some conceptual space, equipped with a metric $d : \mathcal{O} \times \mathcal{O} \to \mathbb{R}^+$. Thus for any two objects $x, y \in \mathcal{X}$ from different classes $c_x \neq c_y$, their distance, in conceptual space, is positive:

$$d(x, y) > 0 \tag{5.8}$$

Formally we have the set of natural objects $\mathcal{O}$, which can be mapped by a distance d to the similarity between any two objects S. Estimating this distance $\hat{d}$ purely from the data, is the central problem in *pattern recognition*. If $\hat{d}$ approximates d sufficiently well then building a classifier is trivial, *e. g.* by using the nearest neighbor classifier.

Let us now assume that instances of these objects are photographed as images, then a natural neural network encodes the object O^i by some highly non-linear transformation f_θ into some representation R. Using accumulated knowledge, the natural neural network can then use the similarity S_{ij} of O^i to some other objects O^j by computing $d(R^i, R^j)$, for further processing[2].

An artificial neural network first cannot directly observe f_θ or d, but must estimate both. Equation 5.9 illustrates the situation visually.

$$\begin{array}{ccc} O & \xrightarrow{d} & S \\ {\scriptstyle \hat{f}_\theta}\downarrow & & \downarrow{\scriptstyle g} \\ \hat{R} & \xrightarrow[\hat{d}]{} & \hat{S} \end{array} \tag{5.9}$$

In our interpretation, the *open-space risk hypothesis* argues theoretically that the issue lies with decision boundary and possibly with the estimated distance function $\hat{d}$, which underestimates the distances to unseen unrelated samples compared to d (Scheirer et al., 2014). However, multiple empirical observations indicate that the samples of unrelated classes are misclassified mainly not because they are too close to the decision boundary (underestimated d), but more severely, unrelated samples end up mostly *within* the decision boundary (Dhamija et al., 2018; Dhamija, 2022; Palechor et al., 2023). To our knowledge, it is not widely known why deep features overlap and how neural networks can be constructed in such a way that they do not overlap. We can, however speculate on the origin of such an overlap. In the formalism shown in Equation 5.9, a feature overlap could be caused by an inconsistent estimation of $\hat{f}_\theta$. Overlapping deep features means that there exist multiple different inputs O^i and O^j which are projected to the same point in deep feature space or less strictly to some ϵ-neighborhood of a different input $R^i \approx R^j$. But when these inputs come from unrelated classes $c_i \neq c_i$ then this should be reflected in their distance in

[2]For illustrative purpose, this process is greatly simplified and many details are omitted for brevity (Miyashita, 1993; Seger and Miller, 2010).

the deep feature space $d(\hat{f}(X^i), \hat{f}(X^j)) > 0$. Since this issue appears to be more common when the input are visually related it appears that

$$P\left(d(\hat{f}(X^i), \hat{f}(X^j)) < \epsilon\right) \propto \quad I(X^i; X^j) \tag{5.10}$$

This could occur if $\hat{f}$ maps those parts of the input X which are merely highly correlated for a particular class, but not causal for the classification. Think of pictures of cows, where the background is most likely green (Xiao et al., 2020). For deep neural network, it is well known that they have an affinity to pay attention to certain correlated features, which don't necessarily generalize, but work very well in standard benchmarks (Geirhos et al., 2020). We call this second potential explanation the *shortcut hypothesis*.

5.2.1 Deep RBF networks for OSR

RBF networks provide a type of classifier which allows fine control over the decision regions. The radial basis functions can be interpreted as a *receptive field* which only focuses on data close to the center of a RBF. Based on the two previously discussed hypotheses on the *open-set recognition problem*, we can derive some predictions on the performance of *RBF networks*.

According to the *open-space risk hypothesis*, in absence of negative training samples, an *RBF classifier* should have a higher performance compared to a standard Softmax classifier, because it bounds the *open-space risk*. In presence of negative training samples, the standard Softmax classifier needs to be modified *e. g.* with the *Entropic Open-Set Loss*. While this still uses Softmax and thus has unbounded decision regions in theory, due to the extra loss, it is statistically less likely that input samples end up "too far away" in the decision region. Which was the motivation to construct this loss in the first place (Dhamija et al., 2018). In this case it is less clear if *RBF classifier* would show a higher performance than *Entropic Open-Set Loss*. With RQ-3 we will explore these performance differences between standard Softmax based methods and deep networks with RBF based classifier using one basis function per class.

The *shortcut hypothesis* is not directly concerned with the geometry of the decision region. Instead it points to deeper issues with modern neural network methods. It is not obvious how a RBF based classifier with one basis function center per class would improve performance for this cause. However, the following informal argument can be made in the case of a multi-center classifier: in the case of training without negative samples and C classes, where $\hat{f}$ classifies samples $\mathcal{X}_{train} \subset \mathcal{D}_p^{train}$ from the training data distribution with a low empirical loss, $\hat{f}$ possibly performs a contractive mapping to the vicinity of C "points". Let us consider those test inputs from the negative samples $\mathcal{X}_{test} \subset \mathcal{D}_n^{test}$, which share high visual similarity with certain training samples $\mathcal{X}_{train}^i$ such that $I(\mathcal{X}_{train}^i; \mathcal{X}_{test}^j) > \epsilon$, where $I(x; y)$ denotes the mutual information between x and y. Then, but without a formal justification, it is plausible that $\hat{f}$ projects these samples close to one of the C "points", where the *closeness* is dependent on their similarity $I(x; y)$ and $\hat{f}$.

Now if a particular class exhibits exists a certain internal structure *i. e.* it has a multimodal distribution with M modes and $\hat{f}$ is modified in a certain way that it projects on average to $M \times C$ "points", called $\hat{f}_M$. Without any formal justification, it is then plausible that

$$P\left(d(\hat{f}_M(X^i), \hat{f}_M(X^j)) < \epsilon\right) < P\left(d(\hat{f}(X^i), \hat{f}(X^j)) \leq \epsilon\right) \tag{5.11}$$

In other words, the probability that two inputs X^i and X^j from distinct classes, but with some visual similarity are projected close to each other in the deep feature representation should be lower in the case of $\hat{f}_M$. However, this depends on more general properties of $\hat{f}_M$. Thus it is not guaranteed that simply projecting the deep features to more than one point per class, will result in

improved *open-set recognition* performance. Nonetheless, we explore networks with this property with RQ-4.

Chapter 6

Approach

In a first step Radial basis functions will be incorporated as a layer in a deep neural network. For a start, only the image domain is considered. A very popular neural network architecture in this domain are the *Neocognitrons* also known as convolutional neural networks. There are multiple ways to incorporate RBFs in a neural networks, thus we explore different configurations.

These variants will be evaluated first in closed-set image classification tasks to compare and investigate their general learning behavior.

In a second step, neural networks with RBF layers will be evaluated in open set recognition tasks. Based on the idea of open-set risk minimization (Scheirer et al., 2013), we expect an RBF network to perform better in an open set recognition task than a baseline neural network.

Since open set recognition specifically emphasized the need to train a model on *negative* classes, different approaches to allow training with negative samples will be investigated. Training with negatives will allow a better evaluation of the RBF networks in open set recognition tasks.

6.1 RBF Layer

In general a RBF layer consists of K RBF units RBF_θ. Each unit processes the whole output of the previous layer or if placed in the beginning, of the input. In this work we will only consider a RBF layer placed as the last layer in a deep network, thus replacing the standard Softmax classification layer. Let us denote the output of the previous layer as $\mathbf{z} \in \mathbb{R}^d$, where d is the dimensionality of that layer. We use the Euclidean distance function δ_{Euc} to compare $\mathbf{z}$ to the *center* μ of the RBF unit.

$$a = \frac{1}{\sqrt{2d}} \delta_{Euc}(\mathbf{z}, \mu) \tag{6.1}$$

To prevent numerical issues with very large distances in high dimensions we scale the resulting Euclidean distance with $\sqrt{2d}$. As the activation function we use the standard Gaussian function parametrized with σ such that it allows for an intuitive interpretation of σ as in the normal distribution.

$$\psi(a) = \exp\left(-\frac{1}{2}\frac{a^2}{\sigma^2}\right) \tag{6.2}$$

Since we have selected the distance function δ and the activation function ψ our RBF unit has two free parameters left μ and σ. Its output can be interpreted as a probability in the same sense as Softmax outputs a probability.

$$RBF_\theta(\mathbf{x}) = \psi(\delta_{Euc}(\mathbf{x}, \mu); \sigma) = \exp\left(-\frac{\delta_{Euc}(\mathbf{z}, \mu)^2}{4d\sigma^2}\right) \tag{6.3}$$

6.1.1 Width parametrization

The parameter σ controls the width of the *receptive field* of an RBF unit. This parameter can be fixed or learned. The dimensionality of it determines the flexibility of the RBF unit to fit data. This also means that its dimensionality is related to the risk of overfitting. In preliminary experiments we used one shared $\sigma \in \mathbb{R}$ for all K RBF units in a layer, but found that using dedicated learnable σ for each unit performs better on our task. σ can also be a vector of dimensionality d or a covariance matrix.

6.1.2 Multilabel classification

The categorical cross-entropy loss rests on the assumptions that all classes are known in terms of the classification. Which can be seen from its formulation as the ratio of the logit of the correct class to the sum of logits from all other possible classes. A generalization of multi-class classification is called *multi-label* classification, where this assumption is dropped and we are no longer required to know all possible classes. However, *multi-label* classification additionally does not assume that only one class or label must be assigned to a sample. In the context of open set classification, we therefore need a slightly constrained *multi-label* classification scheme, where only one label can be assigned, but we do not claim to know the number of all possible labels.

An reasonable approach to model the *multi-label* classification with K labels is to treat each label as an independent binary classification, which is called *One-vs-rest*. This formulation allows the use of a binary cross-entropy loss. In general the likelihood function is, where $t_n \in \{0, 1\}$ is the correct label for class n, y_n is the predicted label:

$$\prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n} \tag{6.4}$$

Which can be maximized by minimizing its negative logarithm:

$$\mathcal{J}_{BCE} = -\sum_{n=1}^{N} (t_n \ln(y_n) + (1 - t_n) \ln(1 - y_n)) \tag{6.5}$$

One issue with this method, is that the loss is imbalanced with respect to a particular target class (Bishop, 2006, p. 338). The information from the true class is compared with the other classes and as K gets larger, the contributions from the true class are weighted with only $1/K$, thus only the majority class is learned well.

Alternatively, the loss contribution of the other classes can be reduced by weighting them accordingly:

$$\mathcal{J}_{BCEW} = -\sum_{n=1}^{N} t_n \ln(y_n) + \frac{1}{K-1}(1 - t_n) \ln(1 - y_n) \tag{6.6}$$

However, eventually we want the training to include not only known classes but also a set of negative samples, without necessarily knowing their potential internal class structure. In this case, the loss would be again unbalanced with regard to the negative classes. As a remedy, we propose to use a different loss that only involves the largest logit other than the correct label. This is essentially just a 1-class/one-vs-rest binary cross entropy loss that maximizes the logit of the true class and minimizes the logit of the nearest false class.

$$\mathcal{J}^t_{MaxBCE} = -\ln(y_t) - \ln(\max_{c \in C \setminus t} y_c) \tag{6.7}$$

However, when negative samples are used in training then y_t does not always exist.

$$\mathcal{J}^t_{MaxBCE} = \begin{cases} -\ln(y_t) - \ln(\max_{c \in C \setminus t} y_c) & \text{if } t \in \mathcal{C}_p, \\ -\ln(\max_{c \in C} y_c) & \text{otherwise,} \end{cases} \tag{6.8}$$

Thus the overall loss function can be written as

$$\mathcal{J}_{MaxBCE} = -\sum_{t \in \mathcal{D}_c} \left(ln(y_t) + \ln(\max_{c \in C \setminus t} y_c) \right) - \sum_{t \in \mathcal{D}_n} \ln(\max_{c \in C} y_c) \tag{6.9}$$

For the weighted version we consider imbalances in the number of classes in the positive set $\mathcal{D}_p$ and the negative training set $\mathcal{D}_n$. Let λ denote the weighting factor

$$\lambda = \frac{|\mathcal{D}_p|}{|\mathcal{D}_p| + |\mathcal{D}_n|} \tag{6.10}$$

then we weight the contributions from the wrong class assignment as

$$\mathcal{J}^t_{MaxBCEW} = \begin{cases} -\ln(y_t) - \lambda \ln(\max_{c \in C \setminus t} y_c) & \text{if } t \in \mathcal{C}_p, \\ -\lambda \ln(\max_{c \in C} y_c) & \text{otherwise,} \end{cases} \tag{6.11}$$

6.1.3 Initialization

The proper initialization of the RBF weights is important for training to converge. Specifically the inputs to the RBF should fall onto its receptive field. Additionally it would be helpful if the input is normalized to some range such that the RBF is at least initially sensitive to all *deep features* (Bishop, 1995, p. 299). We initialize the elements of the centers of the RBF units by sampling from a zero centered normal distribution with standard deviation 0.5. Additionally, we constrain the *deep features* to be close to the origin by employing *batch normalization* (Ioffe and Szegedy, 2015). However, unlike to standard use of *batch normalization*, we do not use the affine transformation.

6.2 Architectures

This section is describes the architectures that are used in our experiments. They follow a shared structure to keep most elements equal to improve the utility of our comparison.

6.2.1 Overview

In this work variants of the architecture in Figure 6.1 are used. Generally, they consists of a CNN based feature extraction head, a deep representation part and a classifier part. To distinguish between these variants a naming scheme is used as shown in Table 6.1.

6.2.2 Feature extraction

For the task of feature extraction different CNN backbones are used depending on the problem complexity. Specifically, we use a *LeNet* variant and a *ResNet* variant.

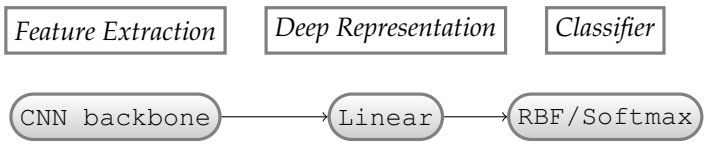


Figure 6.1: MODEL STRUCTURE. *The used models follow the depicted organization structure.*

Table 6.1: ARCHITECTURE VARIANTS NAMING SCHEME. *The naming scheme denotes the three components: feature extraction, deep representation and classifier, separated by an underscore. L denotes a linear fully connected layer.*

Component	Name Part	Description
Feature Extraction	Le	Lenet variant CNN backbone
	Res	Resnet-20 CNN backbone
Deep Representation	L_d	FC linear layer in $\mathbb{R}^d$
	P_a	average pooling layer
	N_b	Batch norm layer
	D	Drop-out layer
Classifier	L_C	Softmax with C classes
	$R_{n;\theta}$	RBF classifier with n basis functions parametrized by θ per class

LeNet variant

This model is a variant of the *LeNet* networks as introduced in Section 3.4.1. Preliminary experiments indicated that two instead of three convolution blocks is sufficient to reach high closed-set classification accuracies on handwriting recognition problems. Our used LeNet variant Le_2 uses therefore only two convolution blocks to increase training speed and reduce overfitting. Additionally, the non-linear activation function has been replaced with the GELU function (Hendrycks and Gimpel, 2016), due to its interesting theoretical properties and increased robustness to noise. Furthermore, dropout is used after each convolution block (Hanson, 1990; Srivastava et al., 2014).

ResNet-20

In this work, the *ResNet-20* variant is used as a feature extraction backbone as introduced in Section 3.4.1. *ResNet-20* contains 20 layers in total, starting with a convolution layer followed by max pooling, then pairs of blocks with different dimensions. Each block H consists of more sub-layers, namely two convolution layers and batch norm in between. At the deep end the network uses average pooling and a fully connected layer of dimensionality depending on the number of output classes. In total, *ResNet-20* has about 3×10^5 trainable parameters. Unfortunately, *PyTorch* does not have a *ResNet-20* implementation, thus the building blocks of their *ResNet-18* implementation was used[1] and adapted accordingly. As a feature extractor the last fully connected layer is omitted and added later as part of the *classifier* as described in the naming scheme 6.1.

[1]From the package *torchvision*, version 0.14.1: https://pytorch.org/vision/0.14/models/generated/torchvision.models.resnet18.html

Table 6.2: ARCHITECTURE VARIANTS LIST. *The naming scheme denotes the three components: feature extraction, deep representation and classifier, separated by an underscore. L denotes a linear fully connected layer.*

	RBF parametrization		
Name	Basis Function	Learnable	Classifier
$Le\_P_aL_{32}D\_L_{10}$	-	-	Softmax
$Res\_P_aL_{32}D\_L_{10}$	-	-	Softmax
$Le\_P_aL_{32}B_nD\_R_{1;G}$	Gaussian	μ, σ	$RBF_{1;\theta}$
$Le\_P_aL_{32}B_nD\_R_{2;G}$	Gaussian	μ, σ	$RBF_{2;\theta}$
$Le\_P_aL_{32}B_nD\_R_{4;G}$	Gaussian	μ, σ	$RBF_{4;\theta}$
$Le\_P_aL_{32}B_nD\_R_{8;G}$	Gaussian	μ, σ	$RBF_{8;\theta}$
$Le\_P_aL_{32}B_nD\_R_{1;W}$	Wide Gaussian with $\rho = 4$	μ, σ	$RBF_{1;\theta}$
$Res\_P_aL_{32}B_nD\_R_{1;G}$	Gaussian	μ, σ	$RBF_{1;\theta}$
$Res\_P_aL_{32}B_nD\_R_{2;G}$	Gaussian	μ, σ	$RBF_{2;\theta}$
$Res\_P_aL_{32}B_nD\_R_{4;G}$	Gaussian	μ, σ	$RBF_{4;\theta}$
$Res\_P_aL_{32}B_nD\_R_{8;G}$	Gaussian	μ, σ	$RBF_{8;\theta}$
$Res\_P_aL_{32}B_nD\_R_{1;W}$	Wide Gaussian with $\rho = 4$	μ, σ	$RBF_{1;\theta}$

6.2.3 Deep Feature Representation

Our *deep feature* layer consists of an *average pooling* layer with kernel size 2×2. Followed by a linear projection to the targeted dimensionality of the *deep feature space* d_{DF}. *Batch normalization* is applied on the resulting representation, but only for RBF models. At the end a *dropout* layer is used for ensembling and reducing overfitting.

6.2.4 Classifier

As the baseline either a standard *Softmax* classifier is used for experimental conditions without negative samples and *Softmax* with Entropic Open-Set Loss (EOS) when using negative samples.

For the RBF baseline the whole *Softmax* layer is replaced with the RBF layer as introduced above. In the subsequent experiments we modify the used RBF units and the ratio of RBF units to the number of classes.

6.3 Exemplar and Prototyp-based Classification

Most machine learning methods can be mapped to one of the two major theories of human *categorization* as described in Section 3.5. Interestingly, the formalism of RBF networks can be mapped to both theories. What needs to be varied is the effective number of RBF centers relative to the number of (known) classes. If they are equal, then such a RBF network corresponds to the *prototype* approaches. Because all the observations from one class are compared to a single condensed representation which thus must contain all the class defining features. On the other hand, if the number of RBF centers is much larger than there are classes, the congruence to the *exemplar* approach is apparent. Furthermore, as an extreme case under some additional restrictions, if the number of RBF centers equals the number of observations, then such a method would be equivalent to the nearest neighborhood rule.

It is not obvious, which approach performs better in general, if any of them even does so. Particularly in the open-set recognition case, it is plausible that an *exemplar* approach could show improved relative performance.

The RBF networks developed in this work, allow the study of both approaches. Whereas the *prototype* approach builds the baseline as it is the "simpler" method.

6.3.1 Exemplar RBF Layer

An *exemplar*-RBF layer adds additional degrees-of-freedom in the design spaces as well as in the parameter space. Regarding the design choices there are multiple aspects that can be varied:

Ratio of centers to classes What is an appropriate number of RBF centers for each known class? The optimal number might be data dependent on the inter-class variances.

Constant Ratio of centers Do all classes require the same number of RBF centers?

Constant RBF types Should all RBF units be of he same type/ parametrization?

To keep the scope of this work bounded, only one variant of *exemplar*-RBF layer will be explored. Specifically a constant multiple m of RBF units per class, all of the same type. Instead, the focus lies on the practicalities of training *exemplar*-RBF layers in the first place. To train such a layer modifications to RBF computation and possibly the loss function are needed. With a functioning *exemplar*-RBF layer, the experiments regarding the RBF image recognition baseline can be repeated with varying multiples of RBFs per class.

Let us denote a multi-center RBF unit as $MRBF_\theta$, containing m regular RBF units. Then the output of this unit is

$$MRBF_\theta(x) = \max_m RBF_\theta^m(x) \tag{6.12}$$

We propose to use the $\mathcal{J}_{MaxBCEW}$ loss function, by taking the maximum over all RBF units. A potential problem with multiple centers per class is that all samples, in the deep feature space, "move" to the initially closest RBF unit, because the samples from the same class usually exhibit high visual correlation and are thus exciting a similar response in the CNN feature extractor.

As a remedy we propose a regularization which maximizes the entropy of the outputs of the regular RBF units for each multi-center RBF unit. For easier implementation, we employ a regularization by using the following extra loss, for K multi-center RBF units containing each m regular RBF units. Let Y be the distribution of outputs from all $K \times m$ RBF units and $H(\cdot)$ denotes the Shannon-Entropy.

$$\mathcal{J}_{ENT} = \mathbb{E}\left[H\left(Y\right)\right] \tag{6.13}$$

6.4 Deep Feature concentration

Deep features are projected to a very tiny subset of available representation space, because the final classification layer functions by calculating the distance to some prototype vector. Particularly for the RBF variants used in this work with Gaussian activation function. Minimizing on of our proposed loss functions would require the deep features of a class to concentrate in one point at the center of the Gaussian. For reference, Figure 6.2a shows Equation 6.14 with $\sigma = 0.25$, for readability we drop the class index k, a denotes the distance to the class prototype a_k, without the high dimension normalization.

$$\phi_k(a_k) = \phi(a) = e^{-\frac{a^2}{2\sigma^2}} \tag{6.14}$$

We hypothesize that the deep features for such an activation function might be not optimal for the *open-set recognition* case, although the activation drops exponentially with increasing distance to the prototype vector for negative samples, it also does so for the samples belonging to this class. Thus it could be very difficult for such a RBF layer to output very high confidence (*e. g.* $> 95\%$) for the positive samples. Therefore we propose alternative activation functions derived from the Gaussian, ideally according to our hypothesis such a function would have a wider *receptive field*, but still drop off sharply. In (6.15) we construct such a function $\phi^A(a;\tau)$. Let τ be a hyperparameter to control the width of the *receptive field*. This activation function is shown in Figure 6.2 Figure 6.2b.

$$\phi^A(a;\tau) = \frac{\min\left(e^{-\frac{a^2}{2\sigma^2}}, \tau\right)}{\tau} \tag{6.15}$$

However, this derivative of this function (Equation 6.16) in the plateau of the *receptive field* is zero. Thus while an RBF with this activation function can *attract* the deep features of the positive samples close to the center, any negative sample that already projects into the *receptive field* cannot be pushed outside anymore.

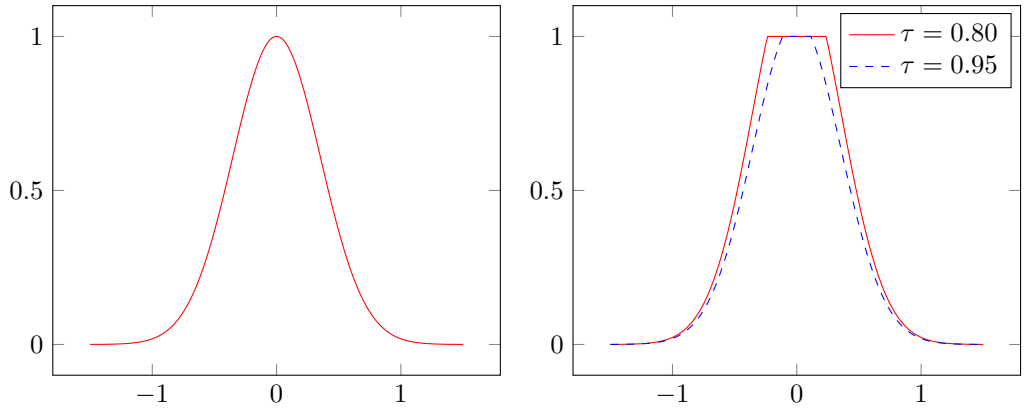
$$\frac{\partial}{\partial a}\phi^A(a;\tau) = \begin{cases} 0 & \text{if } \exp(-x^2/(2\sigma^2)) > \tau \\ -\dfrac{x\exp(-x^2/(2\sigma^2))}{\tau\sigma^2} & \text{otherwise} \end{cases} \tag{6.16}$$

As an approximation of Equation 6.15 we use functions parametrized with $\rho \in \{\, 2n \mid n \in \mathbf{N}_+ \,\}$ of the form

$$\phi^B(a;\rho) = \exp\left(-\frac{a^\rho}{2\sigma^\rho}\right) \tag{6.17}$$

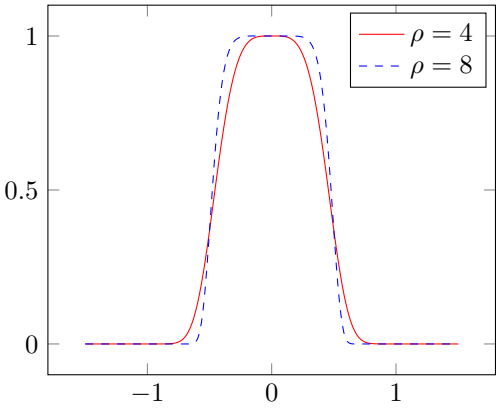
Let us set $\rho = 4$, then the derivative of this activation function (Equation 6.18) is non-zero in the center of the *receptive field*, but still very small and might suffer from saturation effects during training. Figure 6.2c shows an example of this activation function.

$$\frac{\partial}{\partial a}\phi^B(a;\rho=4) = -\frac{2x^3\exp(-x^4/(2\sigma^4))}{\sigma^4} \tag{6.18}$$



(a) Standard Gaussian activation function

(b) Targeted activation function



(c) Wide Gaussian activation function

Figure 6.2: GAUSSIAN ACTIVATION FUNCTION. *Plots of the activation functions with* $\sigma = 0.25$. *Subfigure (a) shows the standard Gaussian, subfigure (b) targeted function and (c) the wide Gaussian activation-*

Chapter 7

Experiments

This chapter gives a description of the exact experimental setups that were performed to answer research questions RQ-3, RQ-4 and RQ-5. The specific configurations are provided in Table 7.1, Table 7.2 and Table 7.3. Each table specifies the training and test dataset, the used loss function and the hyperparameters. The models are optimized with stochastic gradient descent methods, which try to approximate the full gradient of the loss function by taking small *batches* of samples determined by the *batch size*. During each *epoch* all samples from the training data are used once to approximate the gradient. For faster convergence, we use adaptive methods like *Adam* for the handwriting recognition task (Kingma and Ba, 2017). For the natural image recognition task we employ *weight decay* regularization to reduce the impact of overfitting. However, the *Adam* optimizer does not perform well when combined with *weight decay*, instead we use a modified version of *Adam* which improves this combination, called *AdamW* (Loshchilov and Hutter, 2019). Additionally, we employ *Dropout* on the deep features, which is a form of ensembling that greatly improves generalization (Hanson, 1990; Frazier-Logue and Hanson, 2020). For the CIFAR10-50-50 dataset we use data augmentation to reduce overfitting (Chapelle et al., 2000).

7.1 Metrics

For our experiments we use three metrics to measure the performance of the different models. First, the standard classification accuracy, which we denote as the *closed-set accuracy* to emphasize that this metric does not evaluate the *open-set* setting. Second, the Open-Set Classification Rate (OSCR), which does consider the *open-set* setting. Third, a *confidence metric* to shed light on the origins of performance differences as measured by OSCR.

7.1.1 Open-Set Classification Rate

Dhamija et al. (2018) introduced this metric to evaluate the *open-set* aspect of pattern recognition systems. This metric works for recognition systems which give a probabilistic output accompanying a particular class assignment. We denote with $P(c \mid x)$ the estimated probability by the recognition system, that the input x belongs to the class c. During deployment of the system, decision threshold θ must be chosen, according to which the output is either accepted or rejected. Given some operating environment, an imperfect system will always make different types of mistakes, such as erroneously classifying invalid inputs, rejecting valid inputs and miss classifying valid inputs. Depending on the threshold θ some types of mistakes will occur more often that others. The OSCR builds upon this idea and constructs two "sub-metrics". The Correct Classification Rate (CCR) which estimates the accuracy on those inputs which have passed the threshold

θ and are supposed to do so in the first place, because they come from one of the classes that were known to the recognition system during training *i. e.* $\mathcal{D}_p$. The False Positive Rate (FPR) estimates ratio of inputs that were mistakenly not rejected, even though they should have been rejected, because they do not come from any of the known classes *i. e.* $\mathcal{D}_u$. Bisgin et al. (2023) defined these two *rates* formally as follows, let $\hat{c}$ denote the correct class for the input x then

$$CCR(\theta) = \frac{|\{\, x \mid x \in \mathcal{D}_p \wedge \arg\max_c P(c \mid x) = \hat{c} \wedge P(\hat{c} \mid x) \geq \theta \,\}|}{|\mathcal{D}_p|}, \tag{7.1}$$

$$FPR(\theta) = \frac{|\{\, x \mid x \in \mathcal{D}_u \wedge \max_c P(c \mid x) \geq \theta \,\}|}{|\mathcal{D}_u|}, \tag{7.2}$$

These two rates can be estimated for different θ and visualized as a 2D curve with each rate on one axis. To better compare the resulting curves, we will report the CCR corresponding to a particular FPR. To be more precise, since we cannot directly evaluate a specific FPR, we pick a set of evenly spaced points $\{1.0, 0.1, 0.01, 0.001\}$ and select the closest FPR.[1] Then we report the corresponding CCR computed using the same threshold θ. Note that the CCR corresponding to the FPR of 1 is equal with the *closed-set accuracy*.

7.1.2 Confidence metric

The OSCR provides us with an estimate for the performance in the *open-set* environment. We can gain more insights about the differences between two models by looking at potential confounds of the OSCR, namely the average confidence of the model on samples from the *known* and *unknown* classes. For γ^- contains an offset α which is $1/K$ for Softmax classifiers.

$$\gamma^+ = \frac{1}{|\mathcal{D}_p^{test}|} \sum_{i=1 \wedge c_i \in \mathcal{C}_p} y_{i,c_i} \tag{7.3}$$

$$\gamma^- = \frac{1}{|\mathcal{D}_u^{test}|} \sum_{i=1 \wedge c_i \in \mathcal{C}_u} \left(1 - \max_{\mathcal{C}_u} y_{i,c_i} + \alpha\right) \tag{7.4}$$

7.2 Image Recognition Baseline

7.2.1 Handwriting Recognition

In a first set of experiments, we are interested in the openset recognition performance of deep RBF networks on the problem of recognizing handwriting (RQ-3). We are interested in two experimental conditions: one having access to negative samples and one without. The performance of different RBF network configurations is compared to a standard Softmax classifier in the case of of no negative samples and to Entropic-Open set loss in the case with negatives. The models used in these experiments all use the same feature extraction backbone described in subsection 6.2.2. Two baseline models are used in these experiments, $Le\_P_a L_{32} D\_L_{10}$ a Lenet like CNN backbone with a deep feature representation layer of 32 dimensions projected to 10 output dimension and $Le\_P_a L_{32} B_n D\_R_{1;G}$ where the last layer is replaced with 10 Gaussian RBFs each with one learnable σ parameter. These two models are compared using two openset datasets, where the model is trained on a set of positive examples, depending on the experimental condition, additionally

[1] Note that set of FPR that can be estimated, depends on the output distribution of the recognition system. For example, if a system only outputs confidences of either 0 or 1, then only two FPR can be estimated.

Table 7.1: EXPERIMENT CONDITIONS: HANDWRITING RECOGNITION. *Configurations which are only used during training with negative samples are marked with (-)*

	$Le\_P_aL_{32}D\_L_{10}$	$Le\_P_aL_{32}B_nD\_R_{1;G}$
Positive training set	MNIST KMNIST	
Negative training set	First 13 letters of EMNIST	(-)
Test set	Last 13 or 18 letters of EMNIST	
Optimizer	Adam	
Epochs	30	
Learning rate	0.001	
Dropout probability	0.2	
Batch size	128	
Loss	$\mathcal{J}_{CCE}$ / $\mathcal{J}_{EOS}$ (-)	$\mathcal{J}_{MaxBCEW}$

on a set of negative examples and evaluated on a different set of samples. In these experiments the openset datasets Digits-Letters-Letters and Kuzushiji-Letters-Letters are used.

Without negative samples

In the first experimental condition the baseline open-set recognition performance is assessed, when no negative samples are available during training (RQ-3.2). The specific configurations used are listed in Table 7.1.

With negative samples

In the second experimental condition the baseline open-set recognition performance is assessed, when negative samples are available during training (RQ-3.3). In this case the behavior of the model $Le_2\_L_{256}\_L_{10}$ needs to be adjusted, because plain Softmax with categorical cross-entropy loss cannot utilize negative samples. Instead the Entropic Open-Set Loss is used (Dhamija et al., 2018). The specific parameters used are listed in Table 7.1.

7.2.2 Natural Image Recognition

In a second set of experiments, we are interested in the open-set recognition performance of deep RBF networks on the problem of recognizing natural images. Again in the two experimental conditions: one having access to negative samples and one without. The performance of different RBF network configurations is compared to a standard Softmax classifier in the case of of no negative samples and to Entropic-Open set loss in the case with negatives. In these experiments we explore the use of deeper models, particularly *ResNet-20* in addition to the *LeNet* variant. Four models are used in these experiments, $Le\_P_aL_{32}D\_L_{10}$ a Lenet like CNN backbone with a deep feature representation layer of 32 dimensions projected to 10 output dimension and $Res\_P_aL_{32}D\_L_{10}$ where the Lenet CNN backbone is replaced with an *Resnet-20* backbone.

For the RBF variants: $Le\_P_aL_{32}B_nD\_R_{1;G}$ and $Res\_P_aL_{32}B_nD\_R_{1;G}$ where compared the the ones above, the last layer is replaced with 10 Gaussian RBF each with a learnable σ parameter. In these experiments the *open-set* dataset CIFAR10-50-50 is used.

Table 7.2: EXPERIMENT CONDITIONS: NATURAL IMAGE RECOGNITION. *Configurations which are only used during training with negative samples are marked with (-)*

	$Le\_P_aL_{32}D\_L_{10}$ $Res\_P_aL_{32}D\_L_{10}$	$Le\_P_aL_{32}B_nD\_R_{1;G}$ $Res\_P_aL_{32}B_nD\_R_{1;G}$
Positive training set	CIFAR-10	
Negative training set	First 50 classes from CIFAR-100 (-)	
Test set	Last 50 classes of CIFAR-100	
Optimizer	AdamW	
Epochs	160	
Initial Learning rate	0.001	
Dropout probability	0.2	
Weight decay	0.0003	
Augmentations	AutoAugment	
Batch size	128	
Loss	$\mathcal{J}_{CCE}$ / $\mathcal{J}_{CCE}$ (-)	$\mathcal{J}_{MaxBCEW}$
Learning rate schedule	Reduced by one magnitude at epoch 80 and 120	

Without negative samples

In the first experimental condition the baseline open-set natural image recognition performance is assessed, when no negative samples are available during training (RQ-3.4). The specific parameters used are listed in Table 7.2.

With negative samples

Analog to the handwriting recognition task, *Entropic Open-Set Loss* is used for the Softmax variants when training with negative samples. The specific parameters used are listed in Table 7.2.

7.3 Exemplar-based Image Recognition

To investigate whether multiple representation per class provides any advantage (RQ-4), the experiments on handwriting and natural image recognitions are repeated, but with our multi-center RBF model. The experimental parameters are the same as those for the RBF variants described in Table 7.1 and Table 7.2, except for the use of an additional regularization loss $\mathcal{J}_E$ as described in Section 6.3.

7.4 Deconcentrated deep features

Both baseline image recognition experiments are repeated again in both conditions each, but with a different activation function used for the RBF. Specifically, we are interested in the effects of a "wider RBF" on the *open-set recognition* performance, since this would reduce the concentration of deep features at the centers of the RBFs (RQ-5). Table 7.1 and Table 7.2 show the used experimental parameters.

Table 7.3: EXPERIMENT CONDITIONS: LOSS FUNCTION COMPARISON.

	$Le_2\_L_{256}\_R_{10;G(n)}$
Positive training set	MNIST
	KMNIST
Test set	Last 13 letters of EMNIST
Optimizer	Adam
Initial Learning rate	0.001
Dropout probability	0.2
Epochs	30
Batch size	128
Loss	$\mathcal{J}_{BCE}$
	$\mathcal{J}_{BCEW}$
	$\mathcal{J}_{MaxBCE}$
	$\mathcal{J}_{MaxBCEW}$

7.5 Loss functions

An RBF layer as introduced in section 6.1, if placed as the last layer, can replace the usual fully connected layer with Softmax. However, then the standard categorical cross-entropy loss can no longer be used, since it's assumptions are violated. Our research question (RQ-3.1) is concerned with precisely this issue: What loss should be used instead? Multiple loss functions have been discussed theoretically in section 6.1. In a preliminary experiment these loss functions will be experimentally compared among each other: $\mathcal{J}_{BCE}$, $\mathcal{J}_{BCEW}$, $\mathcal{J}_{MaxBCE}$ and $\mathcal{J}_{MaxBCEW}$. In this experiments the *open-set* datasets Digits-Letters-Letters and Kuzushiji-Letters-Letters are used.

Chapter 8

Results

This chapter shows the results of the performed experiments. It is structured by the experimental condition of using negative samples during training. For each condition we will present the results of a Softmax baseline and our equivalent baseline RBF model. Then the RBF variants of using multiple centers for each class and the wider receptive field are presented. At the end, the comparison of the different loss functions is shown. We refer to the CCR at FPR of 1.0 as the *closed-set accuracy*, because this is equivalent to having no rejection threshold.

8.1 OSR without negative samples

8.1.1 Baseline

In Table 8.1 the results of the handwriting recognition experiment are shown for the experimental condition without negative samples. From these results, three observations can be made. First, while the closed-set performance is comparable between the RBF and Softmax variants, the Softmax variant shows slightly higher CCR at a FPR of 1.0. Second, the Softmax variants performed better at lower FPR on both datasets. Third, the average confidence for detecting the positive samples γ^+ is higher for the Softmax variant, whereas the confidence for rejecting the unknown negative samples γ^- is lower compared to the RBF variant. Interestingly, on th Digits-Letters-Letters dataset, we find that the *closed-set accuracy* of both approaches is very similar (± 0.1 percent points), but the confidence γ^+ shows quite a difference. On one hand, this indicates that while both approaches have solved the *closed-set classification* task almost perfectly, the RBF variants are much less confident in their predictions (± 6.7 percent points). On the other hand, this also means that the RBF variants output lower probability estimates for the unknown negative samples and thus show higher confidence in rejecting them γ^-. Alternatively, the Softmax variant might be overconfident and thus falsely accepts more negative samples.

Table 8.2 shows the results for the natural image recognition task on the CIFAR10-50-50 dataset. First, we see that the deeper Resnet shows higher *closed-set accuracy* than the Lenet variants. Overall the Resnet variant trained with standard Softmax shows the best *open-set recognition* performance. The RBF with the Resnet feature extractor shows similar *closed-set accuracy*, but the *open-set recognition* performance drops off rapidly compared to the Softmax variant. The CCR of this RBF variant is only half of the Softmax counterpart at FPR of 0.1 and a tenth at FPR of 0.01. But with the Lenet feature extractor the situation changes. There the RBF variant shows higher *open-set recognition* performance than the Softmax variant, even though it shows slightly lower *closed-set accuracy*. The confidence γ^+ is higher for the Resnets, but γ^- is lower compared to the Lenets, even though the Resnets show better *open-set recognition* performance. Additionally, the RBF variants show higher γ^+ for both feature extractors, but lower γ^- compared to Softmax.

Table 8.1: OSR PERFORMANCE WITHOUT NEGATIVES FOR HANDWRITING RECOGNITION. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ and Softmax to $Le\_P_aL_{32}D\_L_{10}$. Results show average values ($\pm\sigma$) of 5 runs after 30 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Knowns	variant						
KMNIST	RBF	88.5% ±0.3	**33.8% ±0.3**	95.7% ±0.2	85.4% ±0.5	55.7% ±1.3	20.2% ±3.3
	$Softmax$	**95.9% ±0.2**	30.8% ±1.4	**96.4% ±0.1**	**86.1% ±1.5**	**68.4% ±3.7**	**47.2% ±7.1**
MNIST	RBF	92.5% ±0.7	**31.7% ±0.5**	99.3% ±0.0	77.0% ±2.4	29.4% ±5.1	7.0% ±2.7
	$Softmax$	**99.2% ±0.0**	22.4% ±0.4	**99.4% ±0.0**	**82.1% ±1.6**	**49.7% ±6.0**	**25.1% ±5.5**

Table 8.2: OSR PERFORMANCE WITHOUT NEGATIVES FOR NATURAL IMAGE RECOGNITION. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ or $Res\_P_aL_{32}B_nD\_R_{1;G}$ and Softmax to $Le\_P_aL_{32}D\_L_{10}$ or $Res\_P_aL_{32}D\_L_{10}$. Results show average values ($\pm\sigma$) of 5 runs after 160 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
CNN	variant						
Lenet	RBF	**75.0% ±0.3**	32.7% ±0.2	77.5% ±0.7	**46.4% ±0.7**	**21.7% ±0.7**	**4.6% ±1.0**
	Softmax	69.9% ±0.2	**53.2% ±0.3**	**79.6% ±0.2**	46.1% ±0.4	19.4% ±0.6	2.6% ±1.0
Resnet	RBF	**86.8% ±0.1**	19.9% ±0.2	85.6% ±0.2	32.0% ±2.0	3.2% ±0.6	0.4% ±0.2
	Softmax	85.7% ±0.2	**28.7% ±0.3**	**86.7% ±0.2**	**60.1% ±0.6**	**29.0% ±0.8**	**6.3% ±1.3**

8.1.2 Multi-Center RBF

Results for networks with more than one RBF per class are shown in Table 8.3. The table shows results for different numbers of RBFs per class, namely $2, 4$ and 8, for reference the results of the baseline RBF is marked with RBF_1. *Closed-set accuracy* appears to be only marginally affected by multiple RBFs per class, because the values do neither decrease or increase with more centers. However, the highest *closed-set accuracy* is achieved in with 4 and 8 centers, although the difference is very small and within a standard deviation of the other results. Note that in the case of Digits-Letters-Letters, the highest *closed-set accuracy* is the same as the Softmax baseline. It is possible that a ceiling effect occurs for this experiment and a local maximum is reached caused by the used Lenet feature extractor. The differences in *open-set recognition* performance do show an interesting pattern of an inverted U shape. The *open-set recognition* performance first increases with multiple centers of 2 or 4, but decreases again for 8. For both dataset the highest *open-set recognition* performance is achieved by either RBF_2 or RBF_4. This pattern is also visible in terms of the confidence γ^+, which is highest for 2 and/or 4 centers. However, the confidence on the negative unknowns γ^- is lower for the multi-center RBFs, in fact the best performing RBF_2 and RBF_4 show the lowest in each dataset. This indicates that the improved *open-set recognition* performance is possibly due to higher concentration of the samples for each class around their closest RBF center. But this also leads to lower γ^-, however this might not simply be caused due to the increased total number of RBFs, because then we should see a linear correlation between the number of centers and γ^-, instead we see a U-shape.

Table 8.3: OSR PERFORMANCE WITHOUT NEGATIVES FOR HANDWRITING RECOGNITION USING MULTI-CENTER RBFS. *The variants RBF_i refer to $Le\_P_aL_{32}B_nD\_R_{i;G}$. Results show average values ($\pm\sigma$) of 5 runs after 30 epochs.*

Knowns	variant	Confidence γ^+	Confidence γ^-	CCR at FPR 1.0	CCR at FPR 0.1	CCR at FPR 0.01	CCR at FPR 0.001
KMNIST	RBF_1	88.5% ±0.3	**33.8% ±0.3**	95.7% ±0.2	85.4% ±0.5	55.7% ±1.3	20.2% ±3.3
	RBF_2	**91.2% ±0.3**	27.6% ±0.4	95.7% ±0.2	86.8% ±0.8	**63.9% ±4.8**	24.9% ±9.9
	RBF_4	90.5% ±0.5	29.7% ±0.6	95.7% ±0.2	**87.0% ±1.1**	63.3% ±3.8	**25.6% ±3.0**
	RBF_8	89.9% ±0.4	30.8% ±0.4	**95.8% ±0.1**	86.6% ±0.8	58.7% ±5.3	24.0% ±5.9
MNIST	RBF_1	92.5% ±0.7	**31.7% ±0.5**	99.3% ±0.0	77.0% ±2.4	29.4% ±5.1	7.0% ±2.7
	RBF_2	**94.8% ±0.6**	23.7% ±0.2	99.3% ±0.0	**80.7% ±3.4**	**32.4% ±5.9**	8.6% ±3.5
	RBF_4	**94.8% ±0.5**	25.1% ±0.6	**99.4% ±0.0**	77.5% ±3.7	32.0% ±10.7	**9.8% ±7.3**
	RBF_8	92.8% ±0.3	27.2% ±0.6	99.3% ±0.0	73.4% ±5.6	21.0% ±4.9	3.6% ±0.9

For the natural image recognition task, the multi-center results are shown in Table 8.4. The baseline single-center RBF shows the highest *closed-set accuracy* performance with both feature extractors. With the Resnet it also show the higher *open-set recognition* performance. But with the Lenet feature extractor the RBF_2 shows slightly better *open-set recognition* performance.

8.1.3 Wider RBF

Table 8.5 shows the results for using a different Gaussian activation function with a wider *receptive field* (Equation 6.17). For reference, the baseline RBF results from Table 8.1 are added. The wider RBF variants show increased *closed-set accuracy*, where again for the case of the Digits-Letters-Letters dataset the same maximum as with Softmax is reached. Regarding the *open-set recognition* performance, the wider RBFs show consistent improvements across all FPR values. Another difference is visible for the confidences: γ^+ is much higher for the wider RBFs but also γ^- is lower. This is partially expected, since the wider RBFs can assign high confidence to more positive samples without having to collapse them at a single point in deep feature space. The lower γ^- indicates that the negative samples are still close to the *receptive field* of the RBFs, which indicates that the deep features of both subsets are highly correlated.

In Table 8.6 the results for the natural image recognition task are shown. In terms of the confidences, we see the same pattern as with the handwriting task, namely that the wider RBF variants shown an increased confidence on the positive samples γ^+ but reduced for γ^- compared to the baseline RBF variants. For the other metrics there is now a difference dependent on the used feature extractor. When using the Lenet feature extractor, the baseline RBF variants show better *closed-set accuracy* and *open-set recognition* performance. In case of the Resnet feature extractor, the ordering is reversed. Note that the Lenet RBF also showed higher *open-set recognition* performance compared to the Softmax baseline as shown in Table 8.2.

Table 8.4: OSR PERFORMANCE WITHOUT NEGATIVES FOR NATURAL IMAGE RECOGNITION USING MULTI-CENTER RBFS. *The variants RBF_i refer to $Le\_P_aL_{32}B_nD\_R_{i;G}$ or $Res\_P_aL_{32}B_nD\_R_{i;G}$. Results show average values ($\pm\sigma$) of 5 runs after 160 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
CNN	variant						
Lenet	RBF_1	**75.0% ±0.3**	32.7% ±0.2	**77.5% ±0.7**	46.4% ±0.7	21.7% ±0.7	4.6% ±1.0
	RBF_2	74.5% ±0.1	33.6% ±0.2	75.4% ±1.9	**46.6% ±0.5**	**22.5% ±0.4**	**6.0% ±1.3**
	RBF_4	74.0% ±0.4	**34.0% ±0.3**	72.7% ±3.4	45.1% ±1.7	21.7% ±1.0	4.8% ±2.0
	RBF_8	74.2% ±0.2	33.7% ±0.2	75.5% ±0.2	45.5% ±0.5	20.8% ±1.0	4.5% ±1.9
Resnet	RBF_1	86.8% ±0.1	**19.9% ±0.2**	**85.6% ±0.2**	**32.0% ±2.0**	**3.2% ±0.6**	**0.4% ±0.2**
	RBF_2	**88.8% ±0.2**	17.2% ±0.3	85.2% ±0.3	28.2% ±1.1	1.7% ±0.3	0.1% ±0.1
	RBF_4	88.4% ±0.2	17.8% ±0.2	85.4% ±0.4	27.4% ±1.4	1.7% ±0.2	0.2% ±0.1
	RBF_8	87.8% ±0.2	18.2% ±0.4	84.9% ±0.3	29.0% ±1.3	1.8% ±0.5	0.1% ±0.1

Table 8.5: OSR PERFORMANCE WITHOUT NEGATIVES FOR HANDWRITING RECOGNITION USING WIDER RBFS. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ and RBF_{wide} to $Le\_P_aL_{32}B_nD\_R_{1;W}$. Results show average values ($\pm\sigma$) of 5 runs after 30 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Knowns	variant						
KMNIST	RBF	88.5% ±0.3	**33.8% ±0.3**	95.7% ±0.2	85.4% ±0.5	55.7% ±1.3	20.2% ±3.3
	RBF_{wide}	**94.4% ±0.1**	27.5% ±0.6	**96.2% ±0.1**	**86.1% ±0.6**	**62.8% ±1.7**	**27.7% ±3.8**
MNIST	RBF	92.5% ±0.7	**31.7% ±0.5**	99.3% ±0.0	77.0% ±2.4	29.4% ±5.1	7.0% ±2.7
	RBF_{wide}	**98.2% ±0.3**	25.3% ±1.2	**99.4% ±0.0**	**80.6% ±3.9**	**33.8% ±5.2**	**8.0% ±4.4**

8.2 OSR with negative samples

8.2.1 Baseline

In Table 8.7 the results of the handwriting recognition experiment are shown for the experimental condition with utilizing negative samples. Similar to the condition without negatives, the confidence on the positive samples γ^+ is higher for the Softmax variant, however also γ^- is higher in this variant. It is interesting to note that almost for all variants the confidences on the negative unknowns γ^- is higher than γ^+, which is in contrast to the results without negatives in Table 8.1. The differences is lower for γ^- than for γ^+. For the Kuzushiji-Letters-Letters dataset the Softmax variant shows higher *closed-set accuracy* and *open-set recognition* performance. For the Digits-Letters-Letters dataset the situation is reversed and the RBF variants show higher performance on both metrics.

Table 8.6: OSR PERFORMANCE WITHOUT NEGATIVES FOR NATURAL IMAGE RECOGNITION USING WIDER RBFS. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ or $Res\_P_aL_{32}B_nD\_R_{1;G}$ and RBF_{wide} to $Le\_P_aL_{32}B_nD\_R_{1;W}$ or $Res\_P_aL_{32}B_nD\_R_{1;W}$. Results show average values ($\pm\sigma$) of 5 runs after 160 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
CNN	variant						
Lenet	RBF	75.0% ±0.3	**32.7% ±0.2**	**77.5% ±0.7**	**46.4% ±0.7**	**21.7% ±0.7**	4.6% ±1.0
	RBF_{wide}	**76.4% ±0.6**	31.2% ±0.3	74.0% ±2.1	41.8% ±1.4	21.0% ±0.5	**6.0% ±1.6**
Resnet	RBF	86.8% ±0.1	**19.9% ±0.2**	85.6% ±0.2	32.0% ±2.0	3.2% ±0.6	0.4% ±0.2
	RBF_{wide}	**88.4% ±0.2**	14.5% ±0.4	**85.9% ±0.3**	**50.8% ±1.8**	**7.9% ±1.1**	**1.1% ±0.5**

Table 8.7: OSR PERFORMANCE WITH NEGATIVES FOR HANDWRITING RECOGNITION. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ and Softmax to $Le\_P_aL_{32}D\_L_{10}$. Results show average values ($\pm\sigma$) of 5 runs after 30 epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Knowns	variant						
KMNIST	RBF	86.5% ±0.3	91.8% ±0.8	95.5% ±0.2	95.4% ±0.1	94.9% ±0.3	89.3% ±2.7
	$Softmax$	**94.9% ±0.3**	**94.6% ±0.9**	**96.1% ±0.2**	**96.1% ±0.2**	**95.9% ±0.2**	**93.1% ±0.5**
MNIST	RBF	86.6% ±1.7	96.0% ±0.9	**99.2% ±0.0**	**99.2% ±0.0**	**99.2% ±0.0**	**94.7% ±1.7**
	$Softmax$	**95.2% ±1.8**	**97.1% ±0.4**	99.0% ±0.1	99.0% ±0.1	98.6% ±0.1	90.4% ±2.9

Table 8.8: OSR PERFORMANCE WITH NEGATIVES FOR NATURAL IMAGE RECOGNITION. *The variant RBF refers to* $Le\_P_aL_{32}B_nD\_R_{1;G}$ *or* $Res\_P_aL_{32}B_nD\_R_{1;G}$ *and EOS to* $Le\_P_aL_{32}D\_L_{10}$ *or* $Res\_P_aL_{32}D\_L_{10}$. *Results show average values* ($\pm\sigma$) *of* 5 *runs after* 160 *epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
CNN	variant						
Lenet	RBF	**64.1%** ±**0.2**	54.1% ±0.2	61.7% ±1.3	45.0% ±0.3	24.3% ±0.8	**5.3%** ±**1.6**
	Softmax	58.7% ±0.4	**75.1%** ±**0.3**	**79.5%** ±**0.2**	**53.7%** ±**0.5**	**26.1%** ±**0.9**	4.8% ±1.1
Resnet	RBF	**80.2%** ±**0.3**	65.6% ±0.7	85.6% ±0.3	66.0% ±0.6	11.5% ±1.0	1.1% ±0.5
	Softmax	79.5% ±0.3	**74.0%** ±**0.4**	**85.7%** ±**0.4**	**68.9%** ±**0.9**	**37.0%** ±**2.6**	**8.8%** ±**1.4**

Table 8.9: OSR PERFORMANCE WITH NEGATIVES FOR HANDWRITING RECOGNITION USING MULTI-CENTER RBFS. *The variants* RBF_i *refer to* $Le\_P_aL_{32}B_nD\_R_{i;G}$. *Results show average values* ($\pm\sigma$) *of* 5 *runs after* 30 *epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Knowns	variant						
KMNIST	RBF_1	86.5% ±0.3	91.8% ±0.8	**95.5%** ±**0.2**	**95.4%** ±**0.1**	**94.9%** ±**0.3**	89.3% ±2.7
	RBF_2	**89.1%** ±**0.6**	89.1% ±1.6	95.2% ±0.1	95.2% ±0.1	94.7% ±0.1	**93.5%** ±**0.3**
	RBF_4	88.0% ±0.7	91.7% ±1.9	95.4% ±0.2	95.3% ±0.2	94.8% ±0.3	92.1% ±0.6
	RBF_8	87.6% ±0.6	**92.5%** ±**2.9**	95.4% ±0.2	95.3% ±0.2	94.7% ±0.3	92.1% ±0.8
MNIST	RBF_1	86.6% ±1.7	**96.0%** ±**0.9**	**99.2%** ±**0.0**	**99.2%** ±**0.0**	**99.2%** ±**0.0**	94.7% ±1.7
	RBF_2	**91.9%** ±**0.8**	93.7% ±1.0	**99.2%** ±**0.1**	**99.2%** ±**0.1**	**99.2%** ±**0.1**	**97.5%** ±**0.7**
	RBF_4	90.7% ±0.9	95.4% ±0.6	**99.2%** ±**0.1**	**99.2%** ±**0.1**	**99.2%** ±**0.1**	97.2% ±0.4
	RBF_8	89.8% ±0.5	95.5% ±0.9	**99.2%** ±**0.0**	**99.2%** ±**0.0**	**99.2%** ±**0.0**	96.6% ±0.7

For the natural image recognition tasks with negatives during training, the results are presented in Table 8.8. The RBF variants show the highest confidence γ^+ for both feature extractors, but lower confidence on γ^-. The Softmax variant shows both higher *closed-set accuracy* and *open-set recognition* performance, except for the Lenet at FPR of 0.001. Note that the *open-set recognition* performance of the RBF variants drops sharply only with the Resnet feature extractor.

8.2.2 Multi-Center RBF

Table 8.9 shows the results for multiple center per class while using negative samples during training for the handwriting recognition task. Again as in the case of no negatives: RBF_2, RBF_4, RBF_8 refer to the variants with two, four or eight RBF centers per class. RBF_1 is in this case the baseline from Table 8.7. For the Digits-Letters-Letters dataset, the CCR is constantly high among all number of centers and only drops at a low FPR of 0.001. Whereas for the Kuzushiji-Letters-Letters dataset the variant with the highest CCR varies slightly at the different FPR points and the *closed-set accuracy* is highest for the baseline RBF without multiple centers. The RBF_2 variant shows the highest performance at FPR of 0.001. The confidence on the positive samples γ^+ is also highest on RBF_2 and then decreases with more centers. The confidence on the negative samples

Table 8.10: OSR PERFORMANCE WITH NEGATIVES FOR NATURAL IMAGE RECOGNITION USING MULTI-CENTER RBFS. *The variant RBF_{16} refers to $Le_2\_L_{256}\_R_{160;G(n)}$, RBF_8 to $Le_2\_L_{256}\_R_{80;G(n)}$ and RBF_{16} to $Le_2\_L_{256}\_R_{160;G(n)}$. Results show average values ($\pm\sigma$) of 5 runs after 5 epochs.*

CNN	variant	Confidence γ^+	γ^-	CCR at FPR 1.0	0.1	0.01	0.001
Lenet	RBF_1	**64.1% ±0.2**	54.1% ±0.2	61.7% ±1.3	45.0% ±0.3	**24.3% ±0.8**	5.3% ±1.6
	RBF_2	63.7% ±0.3	**55.7% ±0.4**	64.6% ±3.5	**47.2% ±1.6**	23.0% ±0.7	4.9% ±1.8
	RBF_4	63.9% ±0.8	54.6% ±0.5	64.9% ±6.9	47.1% ±5.2	23.5% ±1.7	5.5% ±1.3
	RBF_8	63.6% ±0.4	54.1% ±0.1	**65.4% ±5.5**	46.4% ±3.0	23.2% ±1.3	**6.6% ±1.5**
Resnet	RBF_1	80.2% ±0.3	**65.6% ±0.7**	**85.6% ±0.3**	**66.0% ±0.6**	**11.5% ±1.0**	**1.1% ±0.5**
	RBF_2	**81.7% ±0.2**	64.2% ±0.2	84.1% ±0.4	65.9% ±0.3	10.5% ±0.8	0.8% ±0.1
	RBF_4	81.2% ±0.2	64.3% ±0.3	84.0% ±0.3	65.5% ±0.8	9.9% ±0.6	0.8% ±0.5
	RBF_8	80.6% ±0.1	64.1% ±0.6	84.1% ±0.4	64.9% ±0.3	8.0% ±1.2	0.7% ±0.2

γ^- shows a different pattern, where the variant RBF_2 shows actually the lowest values in both datasets.

For the natural image recognition task, the results are shown in Table 8.10. For the Resnet feature extractor, we find that the multi-center RBFs provide no advantage over the baseline in terms of *closed-set accuracy* and *open-set recognition* performance. Only the confidence on the positive samples γ^+ is higher for the multi-center RBF_2. For the Lenet feature extractor, the situation is different. There, multi-center RBFs show highest *closed-set accuracy* and *open-set recognition* performance, except for FPR 0.01. These multi-center RBFs also have higher or equal γ^- compared to the baseline, but not lower γ^+.

8.2.3 Wider RBF

Table 8.11 shows the results for using a different Gaussian activation function with a wider *receptive field* (Equation 6.17). Compared to the baseline results in Table 8.7, the wider RBF variants show better closed-set accuracy and *open-set recognition* performance. The difference is largest for the Kuzushiji-Letters-Letters dataset at lower FPR points. Additionally, γ^+ is much higher for the wider RBFs, which was expected, since the wider RBFs can assign high confidence to more positive samples without having to collapse them at a single point in deep feature space. Regarding γ^-, the two variants differ slightly depending on the dataset.

For the natural image recognition tasks, the results are shown in Table 8.12. Similar to the results of the handwriting task, the RBF_{wide} variant shows increased *open-set recognition* performance, but only for the Resnet feature exctractor. The *open-set recognition* performance of the Lenet variants is much more similar. The wider RBFs also show higher confidence on the positive samples γ^+. However, the *closed-set accuracy* performance is lower for the wider RBFs compared to the baseline.

Table 8.11: OSR PERFORMANCE WITH NEGATIVES FOR HANDWRITING RECOGNITION USING WIDER RBFS. *The variant RBF refers to $Le_2\_L_{16}\_R_{1;G(n;\rho=4)}$ and $RBF_{1;4)}$ to $Le_2\_L_{16}\_R_{1;G(1;\rho=4)}$. Results show average values ($\pm\sigma$) of* 5 *runs after* 10 *epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Knowns	variant						
KMNIST	RBF	86.5% ±0.3	**91.8% ±0.8**	95.5% ±0.2	95.4% ±0.1	94.9% ±0.3	89.3% ±2.7
	RBF_{wide}	**93.7% ±0.4**	91.6% ±1.0	**96.4% ±0.2**	**96.4% ±0.2**	**95.5% ±0.3**	**90.9% ±1.0**
MNIST	RBF	86.6% ±1.7	96.0% ±0.9	99.2% ±0.0	99.2% ±0.0	**99.2% ±0.0**	94.7% ±1.7
	RBF_{wide}	**97.0% ±0.3**	**96.4% ±0.6**	**99.4% ±0.0**	**99.4% ±0.0**	**99.2% ±0.1**	**96.9% ±0.4**

Table 8.12: OSR PERFORMANCE WITH NEGATIVES FOR NATURAL IMAGE RECOGNITION USING WIDER RBFS. *The variant RBF refers to $Le\_P_aL_{32}B_nD\_R_{1;G}$ or $Res\_P_aL_{32}B_nD\_R_{1;G}$ and RBF_{wide} to $Le\_P_aL_{32}B_nD\_R_{1;W}$ or $Res\_P_aL_{32}B_nD\_R_{1;W}$. Results show average values ($\pm\sigma$) of* 5 *runs after* 160 *epochs.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
CNN	variant						
Lenet	RBF	64.1% ±0.2	54.1% ±0.2	**61.7% ±1.3**	**45.0% ±0.3**	**24.3% ±0.8**	**5.3% ±1.6**
	RBF_{wide}	**66.4% ±0.4**	**54.9% ±0.3**	61.4% ±1.2	44.7% ±1.3	21.8% ±1.0	4.1% ±0.6
Resnet	RBF	80.2% ±0.3	**65.6% ±0.7**	**85.6% ±0.3**	66.0% ±0.6	11.5% ±1.0	1.1% ±0.5
	RBF_{wide}	**81.3% ±0.2**	63.4% ±0.7	85.5% ±0.2	**67.0% ±0.8**	**23.6% ±2.1**	**3.8% ±2.0**

Table 8.13: HANDWRITING OSR PERFORMANCE COMPARISON FOR DIFFERENT LOSS FUNCTIONS WITHOUT NEGATIVES. *The RBF variant used is* $Le\_P_aL_{32}B_nD\_R_{1;G}$. *Results show average values* ($\pm\sigma$) *of* 5 *runs after* 30 *epochs.*

		CCR at FPR			
		1.0	0.1	0.01	0.001
Knowns	Loss Function				
KMNIST	$\mathcal{J}_{BCEW}$	95.4% ±0.3	84.4% ±1.3	53.0% ±3.4	14.5% ±7.7
	$\mathcal{J}_{BCE}$	**95.7% ±0.2**	85.5% ±0.5	**60.9% ±4.6**	**24.6% ±4.6**
	$\mathcal{J}_{MaxBCEW}$	95.5% ±0.2	85.6% ±1.1	56.0% ±7.5	19.5% ±9.1
	$\mathcal{J}_{MaxBCE}$	95.6% ±0.1	**85.8% ±0.4**	58.1% ±2.9	21.5% ±4.3
MNIST	$\mathcal{J}_{BCEW}$	99.2% ±0.0	**80.4% ±4.9**	**34.4% ±8.0**	8.6% ±4.7
	$\mathcal{J}_{BCE}$	**99.3% ±0.0**	79.0% ±0.9	32.9% ±4.5	**9.7% ±3.0**
	$\mathcal{J}_{MaxBCEW}$	**99.3% ±0.0**	77.4% ±3.0	30.2% ±6.6	8.0% ±3.7
	$\mathcal{J}_{MaxBCE}$	**99.3% ±0.0**	76.0% ±5.3	30.4% ±4.1	8.8% ±3.7

8.3 Loss functions

The results of the loss function comparison experiment is shown in Table 8.13 for standard training without access to negative samples. For the closed-set accuracy, there are no large differences between the four loss functions. At smaller FPR more variation is present, but only with large differences for the Kuzushiji-Letters-Letters dataset, although the standard deviation increases with smaller FPR. If we look at the highest CCR in both datasets, then it appears that the $\mathcal{J}_{BCE*}$ loss functions show higher performance than the $\mathcal{J}_{Max*}$ ones.

Regarding the other experimental condition, where negative samples are available during training, the results are shown in Table 8.14. Again, the $\mathcal{J}_{BCE}$ loss function shows highest *closed-set* performance. However at lower FPR thresholds, the $\mathcal{J}_{Max*}$ functions appear to perform better. There does not seem to be a clear difference across both datasets between $\mathcal{J}_{MaxBCE}$ and $\mathcal{J}_{MaxBCEW}$ as both show similar performance.

8.4 Aggregated Results

In Table 8.15 we show the aggregated results over all datasets and CNN feature extractors. Overall we see that the specific RBF variants used in this work do not show better *open-set recognition* performance than the standard Softmax. However, note the very large standard deviations for these results.

Table 8.14: HANDWRITING OSR PERFORMANCE COMPARISON FOR DIFFERENT LOSS FUNCTIONS WITH NEGATIVES. *The RBF variant used is* $Le\_P_aL_{32}B_nD\_R_{1;G}$. *Results show average values* ($\pm\sigma$) *of* 5 *runs after* 30 *epochs.*

		CCR at FPR			
		1.0	0.1	0.01	0.001
Knowns	Loss Function				
KMNIST	$\mathcal{J}_{BCEW}$	94.7% ±0.3	94.7% ±0.3	93.3% ±0.4	82.0% ±2.9
	$\mathcal{J}_{BCE}$	**95.5% ±0.2**	**95.4% ±0.2**	93.6% ±0.5	86.4% ±1.0
	$\mathcal{J}_{MaxBCEW}$	95.1% ±0.1	95.1% ±0.1	94.6% ±0.2	**90.0% ±1.1**
	$\mathcal{J}_{MaxBCE}$	95.4% ±0.2	95.3% ±0.2	**94.8% ±0.3**	89.0% ±2.7
MNIST	$\mathcal{J}_{BCEW}$	98.9% ±0.2	98.9% ±0.2	98.1% ±0.3	88.7% ±4.6
	$\mathcal{J}_{BCE}$	**99.3% ±0.0**	**99.2% ±0.0**	98.2% ±0.5	88.6% ±4.2
	$\mathcal{J}_{MaxBCEW}$	99.2% ±0.0	**99.2% ±0.0**	**99.1% ±0.1**	92.1% ±4.5
	$\mathcal{J}_{MaxBCE}$	99.2% ±0.0	**99.2% ±0.0**	**99.1% ±0.0**	**95.2% ±1.0**

Table 8.15: RESULTS AGGREGATED OVER ALL DATASETS AND FEATURE EXTRACTORS . *Softmax refers to EOS in the case of training with negative samples.*

		Confidence		CCR at FPR			
		γ^+	γ^-	1.0	0.1	0.01	0.001
Negatives	Variant						
No	RBF_1	85.7% ±6.7	29.5% ±5.8	89.5% ±8.8	60.2% ±22	27.5% ±20	8.0% ±7.9
	RBF_2	87.3% ±7.9	25.5% ±6.1	88.9% ±9.6	60.6% ±25	30.1% ±23	9.9% ±11
	RBF_4	86.9% ±8.0	26.7% ±6.2	88.3% ±11	59.2% ±25	29.6% ±23	10.1% ±10
	RBF_8	86.2% ±7.3	27.5% ±6.0	88.9% ±9.6	58.6% ±23	25.6% ±21	8.1% ±10.0
	RBF_{wide}	**89.3% ±8.5**	24.7% ±6.4	88.9% ±10	64.8% ±20	31.4% ±21	10.7% ±11
	Softmax	87.7% ±12	**33.8% ±12**	**90.5% ±8.1**	**68.6% ±17**	**41.6% ±20**	**20.3% ±19**
Yes	RBF_1	79.4% ±9.4	76.9% ±18	85.5% ±15	76.4% ±23	57.5% ±41	47.6% ±46
	RBF_2	81.6% ±11	75.7% ±17	85.8% ±14	76.9% ±22	56.9% ±41	49.2% ±48
	RBF_4	81.0% ±11	76.5% ±18	85.9% ±14	76.8% ±22	56.8% ±42	48.9% ±47
	RBF_8	80.4% ±10	76.5% ±18	86.0% ±14	76.4% ±22	56.3% ±42	49.0% ±47
	RBF_{wide}	**84.6% ±12**	76.6% ±18	85.6% ±15	76.8% ±23	60.0% ±38	48.9% ±46
	Softmax	82.1% ±15	**85.2% ±11**	**90.0% ±8.1**	**79.4% ±19**	**64.4% ±34**	**49.3% ±44**

Chapter 9

Discussion

We will start the discussion of our experimental results by briefly revisiting **RQ2** about the confounding factors in *open-set recognition*. We recall the two hypotheses on the major factors for the problem of *open-set recognition*. The *open-space risk hypothesis*, which states that the decision boundary of a classifier should be minimal, while still allowing for generalization outside of the training data (Moya and Hush, 1996; Scheirer et al., 2013, 2014; Bendale and Boult, 2016). Whereas the *shortcut hypothesis* states that the extracted features of the deep neural network are merely well correlated with the task at hand and not those causally related to the input images. Thus completely inputs, which do however share some visual similarity, get all projected in to the neighborhood of the training data and overlap in the deep feature space.

In RQ-3 we asked about the baseline performance of a simple Deep Radial Basis Function Network (DRBFN) with Gaussian activation function, compared to the standard Softmax variants. According to our interpretation of the *open-space risk hypothesis*, *deep RBF networks* should show higher *open-set recognition* performance compared to the standard Softmax based networks, because they limit the *open-space risk* (Scheirer et al., 2014). However, our results show that our baseline RBF network only outperforms the Softmax baseline in the handwriting recognition experiment using the Digits-Letters-Letters and negative samples. If we look at the results aggregated over both recognition tasks and both involved CNN feature extractors in Table 8.15. Then the baseline RBF variants show lower *open-set recognition* performance on average. Although the performance of the RBF variants tracks the one of Softmax closely in most experiments. Notably on the natural image recognition task using a Resnet as feature extractor, the *open-set recognition* performance of the RBF variant drops sharply even though they show very similar *close-set* performance. This is in contrast to the *Lenet* feature extractor, where on this task the *open-set recognition* performance of both variants is not only similar, but the RBF variants is actually slightly better, except for the *close-set* accuracy. The *Lenet*-RBF variant shows lower *closed-set* accuracy, but higher *open-set recognition* performance. At this point we cannot provide an explanation for this difference in performance. All we can do is point out that these two variants differ mostly in their depth and in the use of residual connections as explained in more detail in Section **??**. To gain more insights into the differences of our baseline models, we utilize the *confidence metric* as described in Subsection 7.1.2. In both experimental conditions of utilizing negative samples, the RBF baseline model shows lower average confidence on the known samples γ^+ compared to the Softmax variants. This indicates that the RBF models are less confident in their predictions, even though they show similar *closed-set accuracies* compared to the Softmax variants. Conversely, when training without negatives, but not when using them, the RBF baseline models show higher confidence γ^- in rejecting unknown negative samples. These confidences directly influence the *open-set recognition* performance, because thresholding is used to perform the rejection functionality. Thus if the confidence on the known samples γ^+ are underestimated then the *open-set recognition* performance drops with high FPR values, given the used evaluation framework.

Using the standard Gaussian activation function in the RBF model, a high confidence in the output can only be achieved if the deep features of a samples are very close to the basis function. In RQ-5 we asked whether such a collapse of the deep features is a beneficial property. With (6.17) constructed alternative bell shaped functions which are much wider at the center. From the aggregated results in Table 8.15, we found that this activation function show not only increased confidence γ^+, but also much better *open-set recognition* performance across all experiments compared to the baseline RBF model.

At least with our parametrization of RBF units and our specific implementation, we did not find that minimizing the *open space risk* helps with *open-set recognition* performance. In fact when we look closer at the definition of the *open space risk*, we do think that it is of limited practical utility.

Since Scheirer et al. (2013) formulates the problem of open set recognition explicitly as the minimization of the empirical risk and the so called *open space risk*, we first explored ways to study the *open space risk* by computing it for a certain classifier and empirical data.

They define *open space risk* as:

$$R_{\mathcal{O}}(f) = \frac{\int_{\mathcal{O}} f(x)dx}{\int_{S_O} f(x)dx} \tag{9.1}$$

Where f is a recognition function which is $f(x) = 1$ if a sample represented by a feature vector $x \in \mathbb{R}^d$ belongs to class y. Furthermore, there are two sets S_O and $\mathcal{O}$, where S_O is defined as a ball containing $\mathcal{O}$ and all training samples. $\mathcal{O}$ denotes the part of "open space" which is labeled as any class. As a first step, we would like to find a way to compute the term $R_{\mathcal{O}}(f)$. To compute this numerically we need to define a finite representation.

Let us denote the representation space as $\mathcal{R}$, which is the finite set of floating point numbers, the set of training samples as $\mathcal{X}$, where $\mathcal{X} \subset \mathcal{R}$. Then using the recognition function f, we can denote the positive labeled space as

$$\mathcal{R}_+ = \{x \mid x \in \mathcal{R} \wedge f(x) = 1\} \tag{9.2}$$

Analogically for $\mathcal{X}_+$,

$$\mathcal{X}_+ = \{x \mid x \in \mathcal{X} \wedge f(x) = 1\} \tag{9.3}$$

then we can reformulate O as

$$\mathcal{O} = \mathcal{R}_+ \setminus \mathcal{X} = \mathcal{R}_+ \setminus \mathcal{X}_+ \tag{9.4}$$

And from the definition we have

$$\mathcal{S}_O = \mathcal{O} \cup \mathcal{X} \tag{9.5}$$

thus

$$\mathcal{S}_O = \mathcal{R}_+ \tag{9.6}$$

then the *open space risk* is:

$$
\begin{aligned}
R_{\mathcal{O}}(f) &= \frac{\int_{\mathcal{O}} f(x)dx}{\int_{S_{\mathcal{O}}} f(x)dx} \\
&= \frac{\int_{\mathcal{O}} f(x)dx}{\int_{\mathcal{R}_+} f(x)dx} \\
&= \frac{\int_{\mathcal{R}_+ \setminus \mathcal{X}_+} f(x)dx}{\int_{\mathcal{R}_+} f(x)dx} \\
&= \frac{\int_{\mathcal{R}_+} f(x)dx - \int_{\mathcal{X}_+} f(x)dx}{\int_{\mathcal{R}_+} f(x)dx}
\end{aligned}
$$

Now if $\mathcal{R} = \mathbb{R}$ then

$$R_{\mathcal{O}}(f) = 1$$

In practice we have to use quantized numbers to compute, in this case case where the set $\mathcal{R}$ is finite, then since $f(x) = 1$ on $\mathcal{R}_+$, the *open space risk* can be calculated in terms of cardinality:

$$
\begin{aligned}
R_{\mathcal{O}}(f) &= \frac{\int_{\mathcal{R}_+} f(x)dx - \int_{\mathcal{X}_+} f(x)dx}{\int_{\mathcal{R}_+} f(x)dx} && (9.7) \\
&= \frac{|\mathcal{R}_+| - |\mathcal{X}_+|}{|\mathcal{R}_+|} && (9.8) \\
&= 1 - \frac{|\mathcal{X}_+|}{|\mathcal{R}_+|} && (9.9)
\end{aligned}
$$

If $|\mathcal{X}_+| = |\mathcal{R}_+|$ the *open space risk* is minimal, this formulation of risk is therefore a function of the "sharpness" of the decision function and the number of samples. The absolute quantity of this risk is only meaningful when the number of samples is not much smaller than the cardinality of the open space. Otherwise the *open space risk* is approximately always one: $R_{\mathcal{O}}(f) \approx 1$.

9.1 Limitations

The different model families Softmax and RBF networks were not compared in exactly the same experimental environment. For instance, the RBF networks use the same loss function in both experimental conditions, even though our proposed loss functions might only be adequate in the condition with negatives. It is possible that the RBF models would show higher performance on the aggregated results when using the standard $\mathcal{J}_{BCE}$ loss when training without negatives. Another difference exists in the deep features layer between the Softmax and RBF networks. Batch normalization was used in front of the RBF networks, but not in front of the Softmax networks. Nonetheless, we believe that the presented experiments provide a fair and informative comparison.

Chapter 10

Conclusion

The remarkable performance of modern deep learning models on many benchmarks can be deceiving. Many models only exhibit "human like" performance only in careful laboratory conditions. As soon as these models are exposed to the complex visual scenery outside of the laboratory, they will at some point be confronted with visual stimuli stemming from natural classes that they have never seen during their training in the lab. In this situation a human would be curious and potentially ask peers about this new stimuli. The "thinking machines" however, would pick one of those classes it knows about and wrongly proclaim, with complete certainty, that this is the correct answer. Even though the machine has never seen stimuli of this class.

Open-set recognition illuminates this issue by evaluating such models in experimental conditions that try to replicate the effect of encountering new classes of stimuli for the first time. A popular hypothesis in the field states that the cause for this issue lies in the "unbounded open space" of the classifier's *decision region*. To put some bounds on this region, we reached to the archives and fetched an older method, *radial basis functions*, which is known to have bounded decision regions. To evaluate if they can be used to improve *open-set recognition* performance.

We integrated *radial basis functions* into modern deep neural networks with a dedicated *RBF layer*. This layer was used to replace the nowadays standard *Softmax* classification layer. We developed a novel loss function, particularly for training with negative samples in the *open-set setting*. The performance of this deep RBF network was then compared to the standard method of using Softmax, including the EOS extension for training with negative samples on three different *open-set datasets*. We identified a potential shortcoming of RBFs with the standard Gaussian activation function. As a remedy we proposed a modification termed *wider Gaussian RBF*, which showed consistently improved performance over the RBF baseline, but not over the Softmax baseline. Based on theories of human *categorization*, we proposed another modification to follow a more *exemplar-based* approach. These so-called *multi-center RBF units* consist of more than one RBF unit per class. To facilitate training of such an *exemplar-based RBF network*, we proposed a regularizer, which helps in keeping more than one center activated. These *exemplar-based RBF networks* show improved performance over the RBF baseline, but not as consistent as the *wider Gaussian RBF*. Furthermore, we put these results into context with the original motivation of this work, namely to bound and minimize the *open-space risk*, which is a widely accepted cause of the above mentioned issues in *open-set recognition*. Given that the studied RBF networks did not outperform the Softmax baseline, we question the validity of the *open-space risk* hypothesis. Instead we think the cause of the *open-set recognition* problem lies in the *shortcut hypothesis* and probably requires some larger changes to the standard deep learning architecture to ever lead to *machines* that can *think properly*.

10.1 Future work

While the specific deep RBF networks studied in this work did not outperform standard Softmax based methods on the used datasets. It is important to remember that we have so far explored only a tiny fraction of the design space of deep RBF networks. Furthermore some techniques show nuanced effects, dependent on the availability of negative samples during training. Thus more work is required to first disentangle all involved effects, before constructing high-performance *open-set recognition* systems.

- To better understand the factors that determine the *open-set recognition* performance of the deep RBF networks studied in this work, it might be of interest to compare the two CNN feature extractors in a way that isolates their differences. This could be done with an experiment that increases the number of layers of the *Lenet* such that it matches those of the *Resnet*. Thus there would be one difference less between the two and the same natural image recognition experiment can be repeated, with the question of whether the residual connections are related to the complex *open-set recognition* performance profile of the deep RBF networks.
- Another promising direction is to potentially improve the performance of the RBFs by initializing the centers in a better way. One could first pre-train a deep network using standard Softmax and then place the RBF centers accordingly (Amirian and Schwenker, 2020).
- Our comparison of the used loss functions indicate that the our used loss function $\mathcal{J}_{MaxBCEW}$ might not be optimal in the condition of training without negative samples. A comparison of the $\mathcal{J}_{MaxBCE}$ loss on larger datasets would be interesting.
- Furthermore, since we showed that our wider RBF unit showed improved performance over the baseline RBF networks and the multi-center RBF networks showed improvements in some cases, it might be of interest to combine the two approaches. A new multi-center RBF layer could be created either consisting of only wide Gaussian RBFs or a mixture of standard and wide RBFs per class.
- It would be obviously interesting to investigate different distance and activation functions. In particular the family of *Weibull kernels*, since they have also successfully been used in *open-set recognition* (Bendale and Boult, 2016).
- While some form of normalization in the deep feature space can be beneficial for the RBF networks like *batch normalization* or *layer normalization* (Ioffe and Szegedy, 2015; Ba et al., 2016). It is unclear whether these normalizations are useful in all conditions (negative sample availability) and during all phases of training.
- RBFs with Gaussian activation function could technically be trained with the full covariance matrix for the shape parameter σ.
- Increasing the number of free parameters in σ indicated in preliminary experiments a potentially increased risk for overfitting. Thus additional data augmentation might be employed as a remedy to artificially increase the training data. A modification of *manifold mixup* could be used (Verma et al., 2019). Specifically, the mixup case of $\lambda = 0.5$ could be treated as a negative sample for both involved classes.

Abbreviations

CCR Correct Classification Rate. 43, 44, 49, 53, 55

CIFAR Canadian Institute For Advanced Research. 20, 25, 26

CNN Convolutional Neural Network. 16, 19, 20, 29, 40, 56, 59, 64

DRBFN Deep Radial Basis Function Network. 59

EMNIST Extended Modified NIST. 24, 26

EOS Entropic Open-Set Loss. 39, 57, 63

FPR False Positive Rate. 44, 49, 51–55, 59

GPU Graphics Processing Unit. 5, 18

KMNIST Kuzushiji-MNIST. 24, 26

MNIST Modified NIST. 16, 19–21, 23, 24, 26

NIST National Institute of Standards and Technology. 23, 24

OCR Optical Character Recognition. 6

OSCR Open-Set Classification Rate. 43, 44

List of Figures

List of Tables

Bibliography

Amari, S. (1967). A Theory of Adaptive Pattern Classifiers. *IEEE Transactions on Electronic Computers*, EC-16(3):299–307.

Amirian, M. and Schwenker, F. (2020). Radial Basis Function Networks for Convolutional Neural Networks to Learn Similarity Distance Metric and Improve Interpretability. *IEEE Access*, 8:123087–123097.

Ansuini, A., Laio, A., Macke, J. H., and Zoccolan, D. (2019). Intrinsic dimension of data representations in deep neural networks. *Neural Information Processing Systems (*.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization.

Barlow, H. B. (1989). Finding Minimum Entropy Codes. *Neural Computation*, 1(3):412–423.

Bartlett, P. L. and Wegkamp, M. H. (2008). Classification with a Reject Option using a Hinge Loss. *Journal of Machine Learning Research*.

Beckman, R. J. and Cook, R. D. (1983). Outlier .... s. *Technometrics*, 25(2):119–149.

Beery, S., Van Horn, G., and Perona, P. (2018). Recognition in Terra Incognita. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *Computer Vision – ECCV 2018*, volume 11220, pages 472–489. Springer International Publishing, Cham.

Bendale, A. and Boult, T. E. (2016). Towards Open Set Deep Networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1563–1572, Las Vegas, NV, USA. IEEE.

Bisgin, H., Palechor, A., Suter, M., and Gunther, M. (2023). Large-Scale Evaluation of Open-Set Image Classification Techniques.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford university press.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

Blanchard, G., Kawanabe, M., Sugiyama, M., and Spokoiny, V. (2006). In Search of Non-Gaussian Components of a High-Dimensional Distribution. *Journal of Machine Learning Research*, 7.

Bonner, R. E. (1966). Pattern Recognition with Three Added Requirements. *IEEE Transactions on Electronic Computers*, EC-15(5):770–781.

Bottou, L., Cortes, C., Denker, J., Drucker, H., Guyon, I., Jackel, L., LeCun, Y., Muller, U., Sackinger, E., Simard, P., and Vapnik, V. (1994). Comparison of classifier methods: A case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition (Cat. No.94CH3440-5)*, volume 2, pages 77–82, Jerusalem, Israel. IEEE Comput. Soc. Press.

Bridle, J. S. (1990). Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition. In Soulié, F. F. and Hérault, J., editors, *Neurocomputing*, pages 227–236. Springer Berlin Heidelberg, Berlin, Heidelberg.

Broomhead, D. S. and Lowe, D. (1988). Multivariable Functional Interpolation and Adaptive Networks. *Complex Systems*, 2(321-355).

Cayton, L. (2005). Algorithms for manifold learning.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58.

Chapelle, O., Weston, J., Bottou, L., and Vapnik, V. (2000). Vicinal Risk Minimization. *Advances in neural information processing systems*.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. *Neural Information Processing Systems*.

Chow, C. (1970). On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46.

Chow, C. K. (1957). An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, EC-6(4):247–254.

Cireşan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column Deep Neural Networks for Image Classification.

Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. (2010). Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. *Neural Computation*, 22(12):3207–3220.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. (2018). Deep Learning for Classical Japanese Literature.

Cohen, G., Afshar, S., Tapson, J., and van Schaik, A. (2017). EMNIST: An extension of MNIST to handwritten letters.

Cortes, C., DeSalvo, G., and Mohri, M. (2016). Learning with Rejection. In Ortner, R., Simon, H. U., and Zilles, S., editors, *Algorithmic Learning Theory*, volume 9925, pages 67–82. Springer International Publishing, Cham.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Costa, J. A. and Hero, A. O. (2006). Determining Intrinsic Dimension and Entropy of High-Dimensional Shape Spaces. In Bellomo, N., Krim, H., and Yezzi, A., editors, *Statistics and Analysis of Shapes*, pages 231–252. Birkhäuser Boston, Boston, MA.

Dechter, R. (1986). Learning While Searching in Constraint-Satisfaction-Problems. *AAAI'86: Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, pages 178–183.

Devroye, L., Györfi, L., and Lugosi, G. (1997). *A Probabilistic Theory of Pattern Recognition*. Stochastic Modelling and Applied Probability. Springer New York.

Dhamija, A. R. (2022). *Five Roads to Open-Set Recognition*. PhD thesis, University of Colorado, Colorado Springs.

Dhamija, A. R., Günther, M., and Boult, T. (2018). Reducing Network Agnostophobia. *Advances in Neural Information Processing Systems*, 31.

Dhamija, A. R., Gunther, M., Ventura, J., and Boult, T. E. (2020). The Overlooked Elephant of Object Detection: Open Set. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1010–1019, Snowmass Village, CO, USA. IEEE.

Duda, R. (1970). 1 Elements of Pattern Recognition. In *Mathematics in Science and Engineering*, volume 66, pages 3–33. Elsevier.

Edgeworth, F. (1887). On discordant observations. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 23(143):364–375.

El-Yaniv, R. and Wiener, Y. (2010). On the Foundations of Noise-free Selective Classification.

Fayin Li and Wechsler, H. (2005). Open set face recognition using transduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1686–1697.

Fefferman, C., Mitter, S., and Narayanan, H. (2016). Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049.

Frazier-Logue, N. and Hanson, S. J. (2020). The Stochastic Delta Rule: Faster and More Accurate Deep Learning Through Adaptive Weight Noise. *Neural Computation*, 32(5):1018–1032.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Ghosh, J. and Nag, A. (2001). An Overview of Radial Basis Function Networks. In Kacprzyk, J., Howlett, R. J., and Jain, L. C., editors, *Radial Basis Function Networks 2*, volume 67, pages 1–36. Physica-Verlag HD, Heidelberg.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Adaptive Computation and Machine Learning Series. MIT Press.

Grassberger, P. and Procaccia, I. (1983). Measuring the strangeness of strange attractors. *Physica D Nonlinear Phenomena*, 9:189–208.

Greff, K., Srivastava, R. K., and Schmidhuber, J. (2017). Highway and Residual Networks learn Unrolled Iterative Estimation.

Grother, P. J. and Hanaoka, K. K. (1995). NIST Special Database 19: Handprinted forms and characters database.

Hanson, S. J. (1990). A stochastic version of the delta rule. *Physica D: Nonlinear Phenomena*, 42(1-3):265–272.

He, K., Zhang, X., Ren, S., and Sun, J. (2015a). Deep Residual Learning for Image Recognition.

He, K., Zhang, X., Ren, S., and Sun, J. (2015b). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification.

Hein, M. and Audibert, J.-Y. (2005). Intrinsic dimensionality estimation of submanifolds in R ^{d}. In *Proceedings of the 22nd International Conference on Machine Learning - ICML '05*, pages 289–296, Bonn, Germany. ACM Press.

Hellman, M. E. (1970). The Nearest Neighbor Classification Rule with a Reject Option. *IEEE Transactions on Systems Science and Cybernetics*, 6(3):179–185.

Hendrickx, K., Perini, L., Van der Plas, D., Meert, W., and Davis, J. (2021). Machine Learning with a Reject Option: A survey.

Hendrycks, D. and Gimpel, K. (2016). Gaussian Error Linear Units (GELUs).

Hendrycks, D. and Gimpel, K. (2017). A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In *ICLR*.

Herbei, R. and Wegkamp, M. H. (2006). Classification with reject option. *Canadian Journal of Statistics*, 34(4):709–721.

Hochreiter, S. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hodge, V. J. and Austin, J. (2004). A Survey of Outlier Detection Methodologies.

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1):106–154.

Hüllermeier, E. and Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506.

Ide, E. R. and Tunis, C. J. (1967). An Experimental Investigation of a Nonsupervised Adaptive Algorithm. *IEEE Transactions on Electronic Computers*, EC-16(6):860–864.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

Ivakhnenko, A. G. (1971). Polynomial Theory of Complex Systems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-1(4):364–378.

Jäkel, F., Schölkopf, B., and Wichmann, F. A. (2008). Generalization and similarity in exemplar models of categorization: Insights from machine learning. *Psychonomic Bulletin & Review*, 15(2):256–271.

Kanal, L. (1974). Patterns in pattern recognition: 1968-1974. *IEEE Transactions on Information Theory*, 20(6):697–722.

Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization.

Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (Nov./1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

LeCun, Y., Jackel, L. D., Bottou, L., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Muller, U. A., Sackinger, E., Simard, P., Vapnik, V., and Laboratories, T. B. (1995). LEARNING ALGORITHMS FOR CLASSIFICATION: A COMPARISON ON HANDWRITTEN DIGIT RECOGNITION. *Neural networks: the statistical mechanics perspective*, 261(276).

Lee, A. B., Pedersen, K. S., and Mumford, D. (2003). The Nonlinear Statistics of High-Contrast Patches in Natural Images.

Linnainmaa, S. (1970). The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. Master's thesis, University of Helsinki, Helsinki, Finland.

Loshchilov, I. and Hutter, F. (2019). Decoupled Weight Decay Regularization. In *ICLR 2019*.

Majewski, W. and Basztura, C. (1984). Speaker Recognition in Open Sets. In Cohen, A. and Broecke, M. P. R. V. D., editors, *Proceedings of the Tenth International Congress of Phonetic Sciences*, pages 322–325. De Gruyter Mouton.

Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

Minsky, M. (1961). Steps toward Artificial Intelligence. *Proceedings of the IRE*, 49(1):8–30.

Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. The MIT Press. MIT Press.

Miyashita, Y. (1993). Inferior Temporal Cortex: Where Visual Perception Meets Memory. *Annual Review of Neuroscience*, 16:245–263.

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the Number of Linear Regions of Deep Neural Networks. *Advances in neural information processing systems*.

Mori, S., Suen, C., and Yamamoto, K. (1992). Historical review of OCR research and development. *Proceedings of the IEEE*, 80(7):1029–1058.

Moya, M. M. and Hush, D. R. (1996). Network constraints and multi-objective optimization for one-class classification. *Neural Networks*, 9(3):463–474.

Nagy, G. (1968). State of the art in pattern recognition. *Proceedings of the IEEE*, 56(5):836–863.

Narayanan, H. and Mitter, S. (2010). Sample Complexity of Testing the Manifold Hypothesis. *NIPS'10: Proceedings of the 23rd International Conference on Neural Information Processing Systems*, 2.

Northcutt, C. G., Athalye, A., and Mueller, J. (2021). Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks.

Palechor, A., Bhoumik, A., and Gunther, M. (2023). Large-Scale Open-Set Classification Protocols for ImageNet. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 42–51, Waikoloa, HI, USA. IEEE.

Park, J. and Sandberg, I. W. (1991). Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation*, 3(2):246–257.

Pascanu, R., Montufar, G., and Bengio, Y. (2014). On the number of response regions of deep feed forward networks with piece-wise linear activations.

Pestov, V. (2007). An axiomatic approach to intrinsic dimension of a dataset.

Pimentel, M. A., Clifton, D. A., Clifton, L., and Tarassenko, L. (2014). A review of novelty detection. *Signal Processing*, 99:215–249.

Pineda-Arango, S., Obando-Paniagua, D., Dedeoglu, A., Kurzendörfer, P., Schestag, F., and Scholz, R. (2020). Improving Sample Efficiency with Normalized RBF Kernels.

Pope, P., Zhu, C., Abdelkader, A., Goldblum, M., and Goldstein, T. (2021). The Intrinsic Dimension of Images and Its Impact on Learning.

Roady, R., Hayes, T. L., Kemker, R., Gonzales, A., and Kanan, C. (2020). Are Out-of-Distribution Detection Methods Effective on Large-Scale Datasets? *PLOS ONE*, 15(9):e0238302.

Robbins, H. and Monro, S. (1951). A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Cornell Aeronautical Laboratory. Report No. VG-1196-G-8. Spartan Books.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge.

Scheirer, W. J., de Rezende Rocha, A., Sapkota, A., and Boult, T. E. (2013). Toward Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1757–1772.

Scheirer, W. J., Jain, L. P., and Boult, T. E. (2014). Probability Models for Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2317–2324.

Schmidhuber, J. (1992). Learning Factorial Codes by Predictability Minimization. *Neural Computation*, 4(6):863–879.

Schmidhuber, J. (2015). Deep Learning in neural networks: An overview. *Neural Networks*, 61:85–117.

Seger, C. A. and Miller, E. K. (2010). Category learning in the brain. *Annual Review of Neuroscience*, 33:203–19.

Sevilla, J., Heim, L., Ho, A., Besiroglu, T., Hobbhahn, M., and Villalobos, P. (2022). Compute Trends Across Three Eras of Machine Learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Padua, Italy. IEEE.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

Simard, P., Steinkraus, D., and Platt, J. (2003). Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, volume 1, pages 958–963, Edinburgh, UK. IEEE Comput. Soc.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15.

Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015a). Highway Networks.

Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015b). Training Very Deep Networks.

Steinbuch, K. (1961). Die Lernmatrix. *Kybernetik*, 1(1):36–45.

Tabernik, D. (2021). *Representing Visual Entities with Deep Hierarchical and Compositional Models*. PhD thesis, University of Ljubljana, Ljubljana.

Tabernik, D., Kristan, M., Wyatt, J. L., and Leonardis, A. (2016). Towards deep compositional networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3470–3475, Cancun. IEEE.

Torralba, A., Fergus, R., and Freeman, W. (2008). 80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, 59(236):433–460.

van Amersfoort, J., Smith, L., Teh, Y. W., and Gal, Y. (2020). Uncertainty Estimation Using a Single Deep Deterministic Neural Network. *Proceedings of the 37th International Conference on Machine Learning*.

Vapnik, V. (1991). Principles of Risk Minimization for Learning Theory. *Advances in neural information processing systems*, 4.

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Courville, A., Lopez-Paz, D., and Bengio, Y. (2019). Manifold Mixup: Better Representations by Interpolating Hidden States.

Walkowiak, T., Szyc, K., and Maciejewski, H. (2021). On Validity of Extreme Value Theory-Based Parametric Models for Out-of-Distribution Detection. In Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V. V., Dongarra, J. J., and Sloot, P. M., editors, *Computational Science – ICCS 2021*, volume 12744, pages 142–155. Springer International Publishing, Cham.

Wen, Y., Zhang, K., Li, Z., and Qiao, Y. (2016). A Discriminative Feature Learning Approach for Deep Face Recognition. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, volume 9911, pages 499–515. Springer International Publishing, Cham.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. (2020). Noise or Signal: The Role of Image Backgrounds in Object Recognition.

Zadeh, P. H., Hosseini, R., and Sra, S. (2018). Deep-RBF Networks Revisited: Robust Classification with Rejection.

Zhang, X., Zhao, R., Qiao, Y., and Li, H. (2020). RBF-Softmax: Learning Deep Representative Prototypes with Radial Basis Function Softmax. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Computer Vision – ECCV 2020*, volume 12371, pages 296–311. Springer International Publishing, Cham.