UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Rain Vagel

Radial Softmax: A Novel Activation Function for Neural Networks to Reduce Overconfidence in Out-Of-Distribution Data

Master's Thesis (30 ECTS)

Supervisor: Ardi Tampuu, MSc
Supervisor: Meelis Kull, PhD
Supervisor: Raul Vicente, PhD

Tartu 2020

Radial Softmax: A Novel Activation Function for Neural Networks to Reduce Overconfidence in Out-Of-Distribution Data

Abstract:
Neural networks are used widely and give state-of-the-art results in fields such as machine translation, image classification and speech recognition. These networks operate under the assumption that they predict on data that originates from the same distribution, as the training data. If this is not the case then the model will output incorrect results, often with very high confidence. In this work we explain how the commonly used softmax is unable to mitigate these problems and propose a new function called radial softmax which might help to mitigate out-of-distribution (OOD) overconfidence issues. We show that radial softmax is capable of mitigating OOD overconfidence issues in almost all cases. Based on our literature review this is the first time an improvement to softmax has been proposed for this issue. We also showed that changes to the training cycle or intermediate activation functions are not needed. With this function it is possible to make the models more resistant to OOD data without modifications to the larger architecture or training cycles. By having models that we know are resistant to OOD data, we can be more confident in the model output and use them for applications where mistakes are unacceptable such as healthcare, the defence industry or autonomous driving.

Keywords: Neural Networks, Machine Learning, Softmax, Out-of-Distribution data, Overconfidence

CERCS: P175, P176

Radial softmax: uus närvivõrkude aktivatsiooni funktsioon, et vähendada liigset kindlust jaotuseväliste andmestike puhul

Lühikokkuvõte:

Tehisnärvivõrke kasutatakse laialdaselt heade tulemustega aladel nagu masintõlge, piltide klassifitseerimine ja häältuvastus. Nende võrkude eelduseks on, et kasutusandmed on treeningandmetega samast jaotusest. Vastasel juhul väljastab mudel kõrge kindlusega valesid tulemusi. Selles töös tutvustame me, kuidas *softmax* selles olukorras käitub ning selle probleemidest. Me tutvustame uut aktivatsiooni funktsiooni nimega *radial softmax*, mis on paremini võimeline ära tundma jaotuseväliseid andmeid. Me esitame tulemusi, mis näitavad, et meie funktsioon annab osade jaotuseväliste andmete puhul *softmax*'ist paremaid tulemusi. Vastavalt kirjanduse ülevaatele oleme me esimesed, kes pakuvad välja *softmax*'ile alternatiivset funktsiooni selle probleemi lahendamiseks. Meie funktsiooniga ei ole vaja teha muudatusi mudelite treenimisel ega teha liigseid muudatusi nende arhitektuurides. Neid mudeleid, mille puhul me teame, et need ei väljasta kõrge kindlusega valesid väljundeid, saame kasutada alades, kus vead pole akepteeritavad.

Nendeks on näiteks meditsiin, riigikaitse ja autonoomne juhtimine.

Võtmesõnad: Masinõpe, Tehisnärvivõrgud, Softmax, jaotusest välised andmed, liigne kindlus

CERCS: P175, P176

Contents

1 Introduction

Deep neural networks have been adapted to a wide variety of applications where they provide state-of-the-art performance, such as image classification and speech recognition [1, 2]. It has been established that neural networks using ReLU activation functions in their hidden layers, which we will call ReLU models or architectures in this work, and softmax for output functions, are unable to differentiate between out-of-distribution (OOD) data and in-distribution data [3, 4, 5]. Both of those activation functions are highly popular and used widely in existing architectures [6]. In-distribution data is any data that the model was trained on or from the same category. OOD data is any other data. So for example if a model is trained on hand drawn digits ranging from 0 to 9, then images of those numbers are in-distribution. However if we would give that model images of horses, then those images would be OOD.

This means that these models make high confidence predictions on unrecognized data [7]. This leads to problems in adopting neural networks in domains where errors are not tolerated such as medicine. Another problematic area of adoption is where the models receive open-domain data while in use such as autonomous driving, where model developers do not have control over the data that it will score.

In this work we focus on softmax, which as a function receives a vector of class scores and outputs a vector of class probabilities [8]. In classification neural networks it is used to turn model outputs into probabilities, from which the highest probability is both the prediction and confidence for said prediction. The definition of softmax can be seen in Equation 1. We see empirically that models using this activation function fail to react well to OOD samples. Our main contribution is in modifying softmax so that it is more equipped to deal appropriately with OOD inputs.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}},\ for\ i = 1, \ldots, K\ and\ z = (z_1, \ldots, z_K) \in \mathbb{R}^{\mathbb{K}} \tag{1}$$

In this work we propose a new activation function called radial softmax. It is based on softmax and is able to learn parameters that would help it limit the maximum confidence for a class and in predicting class balance far from the dataset. The proposed function can be easily used in a model without requiring difficult changes to the architecture or the development of custom training cycles.

Our main contributions in this work are:

1. Explaining and visualising the problems of softmax
2. Defining a novel activation function called radial softmax
3. Comparing models with the new function against baseline models
4. Performing an ablation study

This thesis is broken down into sections as follows. We explain and visualise why softmax is unable to help distinguish OOD samples from in-distribution samples in Section 2. In this section we also define radial softmax and share implementation details in Tensorflow 2.0 [9]. The code is available in our Github repository [1].

The experimental setup, datasets and architectures are described in Section 3. We use toy datasets to validate the effects in a two dimensional input space before comparing radial softmax against baseline models on real-life datasets. The metrics, datasets and architectures used by us have been used in related work, making our results comparable to theirs.

Results are presented in Section 4 with analysis on the performance of radial softmax. We have further done an ablation study to explore the effects in Section 5, where we describe the experiment setup and share results. Based on the results we present future work on Section 6.

Related work and other proposed solutions to the OOD samples distinguishing problem are introduced in Section 7. This encompasses work done on custom training cycles, novel intermediate activation functions, new model architectures etc.

We finish the thesis with a conclusion in Section 8.

[1] Thesis Github repository: `https://github.com/RainVagel/confidence-thesis`

2 Softmax & Radial Softmax

Softmax is one of the most popular output activation functions currently used in neural networks [6]. However as we discuss further in Section 2.1, often it is unable to handle OOD data points, leading to false classifications with extremely high probabilities [5] [4]. The definition can be seen in Equation 2 [8].

In Section 2.1 we discuss the problems and effects of softmax for which we will be proposing a possible solution to in this work. We also formulate the problem and explain the metrics that we will be using to see if the new proposed function works as intended.

In Section 2.2 we propose a novel activation function based on softmax, called radial softmax. The new function is designed to learn class balances and maximum probability vectors, which should make it possible for softmax to make predictions based on learned class balances.

As one of the results of this work, radial softmax is implemented in Tensorflow 2.0, currently one of the most popular deep learning frameworks, as a new layer [9] [10]. This allows easy use with both Tensorflow and Keras frameworks [9, 11]. The implementation with the code attached is in Section 2.3.

2.1 Problems with Softmax

In this work we are interested in the in- and out-of-distribution detection problem. Namely for any data point, is the model able to predict if it is from the in-distribution dataset or OOD dataset?

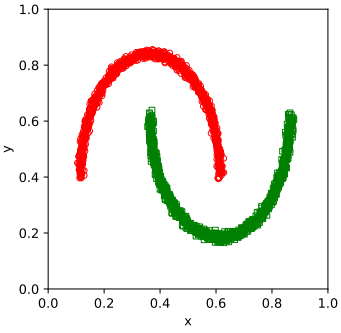
In our work we focus on the softmax function and its role in detecting OOD samples. Softmax is focused on because it is a widely used output activation function [6].

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \ for\ i = 1, \ldots, K\ and\ z = (z_1, \ldots, z_K) \in \mathbb{R}^{\mathbb{K}} \tag{2}$$

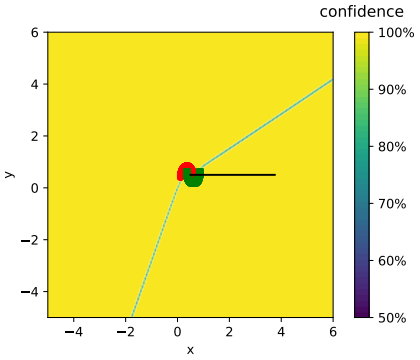
Softmax outputs a vector of class probabilities, $\sigma(z_i) + \ldots + \sigma(z_K) = 1$. Thus each probability is dependent on the other probabilities, specifically how large the z of one class, or the class score, is in comparison to the other class scores. A higher class score in respect to all other scores results in a higher class probability.

Using ReLU models to make predictions on data far from the training data, often results in the difference between class scores to widen, growing large either negatively or positively. This results in the class probability for any of the classes to approach 1, thus making a false overly confident prediction [4]. To visualise this effect we will be training a two hidden layer model with intermediate ReLU activations and a softmax output activation function on the two moons dataset. The two moons dataset consists of 2000 training samples and is made up of two classes, with 1000 training samples per class. It is visualised in Figure 1a. Every data point not in the training set is OOD and

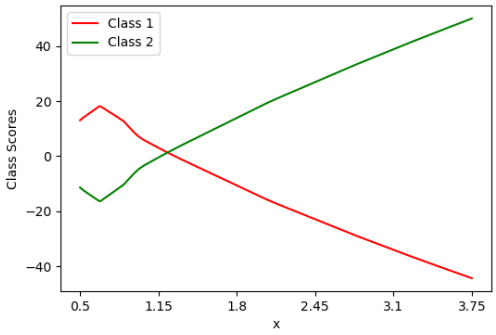
the models respective confidence for them is for the most part 100%, which is visualised in Figure 1b.



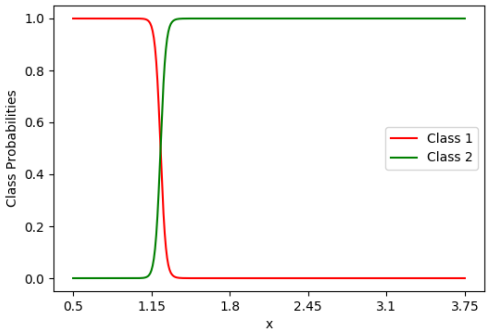
(a) Two moons dataset centered around (0.5, 0.5).



(b) Confidence areas with OOD sample line for softmax model trained on two moons.



(c) Class scores for OOD samples for softmax model trained on two moons.



(d) Class probabilities for OOD samples for softmax model trained on two moons.

Figure 1. Softmax model run on a two moons dataset with two classes, visualised in Figure 1a. The models confidence levels around the training dataset are visualised in Figure 1b with a black line representing where we gather OOD samples from. Class scores and probabilities for the OOD samples are visualised in Figures 1c and 1d respectively.

Figure 1c visualises class scores of data gathered in 0.01 steps from the black line in Figure 1b, starting from between the two moons. After crossing a point where both class scores are 0, class 2 dominates class 1 and the difference is increasing nearly linearly. The class probabilities shown in Figure 1d show that the model is nearly always 100% confident in predicting class 2 even though the data points are far from the training data, as in out-of-distribution. The fact that the model predicts one of the classes with really high confidence in OOD samples means that the model is unable to distinguish between

in-distribution data and OOD data.

2.2 Defining Radial Softmax

We propose a new activation function based on softmax, called radial softmax and have defined it in Equation 3.

$$s(\mathbf{z})_i = \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{j=1}^{K} (e^{c_j - |z_j|} + e^{b_j})} \tag{3}$$

In designing radial softmax we had the following goals:

1. Models trained with $s(\mathbf{z})$ must have comparable performance to models trained with softmax for in-distribution data.

2. For OOD data a model trained with $s(\mathbf{z})$ should output class probabilities that are approximately equal to the class distribution $\pi_1, \ldots, \pi_K$.

Thus $s(\mathbf{z})$ must act based on the class distribution within the training data and be comparable to softmax for in-distribution data.

We establish four *desiderata* for radial softmax:

1. For each Θ, $s(\mathbf{z})$ return a vector of class probabilities. Meaning $0 \leq s(\mathbf{z})_i \leq 1$, for each $i = 1, \ldots, K$ and $\sum_{i=1}^{K} s(\mathbf{z})_i = 1$.

2. The function $s(\mathbf{z})_i$ outputs maximum probability at $z_i = 0$ and smaller probabilities as data farther from the in-distribution dataset is gathered. For each Θ and for each i and for any real values $z_1, \ldots, z_{i-1}, z_{i+1}, \ldots z_K$, the function $s(\mathbf{z})_i$ as a function of z_i is symmetrical around $z_i = 0$, strictly decreasing as $z_i \to \infty$ and maximised at $z_i = 0$.

3. For each class distribution π_i there exists Θ such that if all z_i are far from 0, then $s(\mathbf{z})$ is approximately equal to the class distribution $\pi_1, \ldots, \pi_k$ independent whether z_i have similar values or not. For each $\epsilon > 0$ and for each $z_1, \ldots, z_K$, such that $|z_i| > M$, then it follows $|s(\mathbf{z})_i - \pi_i| < \epsilon$.

4. Maximal achievable probability for class i is M_i, where $\pi_i < M_i < 1$. For each $M_1, \ldots, M_K$ there exists Θ, such that for each i we have that $\sup_{\mathbf{z}} s(\mathbf{z})_i = M_i$.

We define the function as $s(\mathbf{z})_i = \frac{t(z)_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$ and based on that definition we have found that $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ fulfills all of our *desiderata*. We prove it by presenting a lemma and theorems for each of the *desiderata*. All of our proofs are available in Appendix I.

Lemma 1. *Let $c > 0$ and $f(x) = \frac{x}{x+c}$. Then f(x) is strictly increasing in the range $x \in (0, \infty)$. That means $f(x') > f(x)$ for every $x' > x > 0$.*

We next define Theorem 2, which states the basic requirements of softmax.

Theorem 2. *Let $t(\mathbf{z})$ and $s(\mathbf{z})$ be vectors with elements $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$, where $b_1, \ldots, b_k, c_1, \ldots, c_k \in \mathbb{R}$. Then $0 \leq s(\mathbf{z})_i \leq 1$, for $i = 1, \ldots, K$ and $\sum_{i=1}^{K} s(\mathbf{z})_i = 1$.*

Since the function returns a vector of class probabilities, each element must be between 0 and 1 and the sum must be equal to 1.

Theorem 3. *Let $t(\mathbf{z})$ and $s(\mathbf{z})$ be vectors with elements $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$, where $b_1, \ldots, b_k, c_1, \ldots, c_k \in \mathbb{R}$. Then for each i and for any real values $z_1, \ldots, z_{i-1}, z_{i+1}, \ldots z_K$, the function $s(\mathbf{z})_i$ as a function of z_i is symmetrical around $z_i = 0$, strictly decreasing as $z_i \to \infty$ and maximised at $z_i = 0$.*

Theorem 3 that the maximum class probability is when the class score is 0 and when data is far from the in-distribution data, the respective class probability decreases. As Theorem 3 stands, radial softmax should return lower probabilities for data far from the training dataset.

Theorem 4. *Let $t(\mathbf{z})$ and $s(\mathbf{z})$ be vectors with elements $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$, where $b_i = \ln \pi_i$ and $c_1, \ldots, c_k \in \mathbb{R}$. Then for $\epsilon > 0$ and for each $z_1, \ldots, z_K$, such that $|z_i| > M$, it follows that $|s(\mathbf{z})_i - \pi_i| < \epsilon$.*

From Theorem 4 we fulfill our second goal to output approximate class distributions for OOD data by having $\mathbf{b}$ be the vector of class distributions. Our implementation of radial softmax allows this parameter to be learnable, thus the user does not need to have prior knowledge of the class distribution.

A function that lacks the class distribution parameter will not be able to meet the requirements of our third *desiderata*.

Theorem 5. *For each* $M_1, M_2, \ldots, M_k \in (0, 1)$*, where* $M_i > \pi_i$*, there exists* $b_1, \ldots, b_K, c_1, \ldots, c_K \in \mathbb{R}$*, such that for each i,* $\sup_{\mathbf{z}} s(\mathbf{z})_i = M_i$*. Where* $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ *and* $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$.

With Theorem 5 we explain our use of the c parameter. This parameter is a vector which defines the maximum class probability for any class. Any function that this theorem can be used for matches our fourth *desiderata* and any that do not have the maximum class probability vector fail.

With this we have shown that radial softmax meets the requirements of all of our *desiderata* and we can continue on to the implementation.

2.3 Implementing Radial Softmax

Radial softmax is implemented in Tensorflow 2.0 as a custom layer. This allows it to be easily used while defining models either with the Tensorflow or Keras API's [9, 11]. The source code can be seen in Listing 1.

```
class RadialSoftmax(Layer):
        def __init__(self, c_initializer=initializers.Zeros(),
            b_initializer=initializers.Zeros(),
                c_trainable=True, b_trainable=True, **kwargs):
        super(RadialSoftmax, self).__init__(**kwargs)
        self.c_initializer = c_initializer
        self.b_initializer = b_initializer
        self.supports_masking = True
        self.c_trainable = c_trainable
        self.b_trainable = b_trainable

    def build(self, input_shape):
        self.c = self.add_weight(name="c",
                                 shape=(input_shape[1],),
                                 initializer=self.c_initializer,
                                 trainable=self.c_trainable)
        self.b = self.add_weight(name="b",
                                 shape=(input_shape[1],),
                                 initializer=self.b_initializer,
                                 trainable=self.b_trainable)
        super(RadialSoftmax, self).build(input_shape)

    def call(self, inputs):
        first_exp = tf.exp(self.c - tf.abs(inputs))

        p = (first_exp + tf.exp(self.b)) / tf.reduce_sum(first_exp +
            tf.exp(self.b), axis=1, keepdims=True)

```

```
27          return p
28
29      def compute_output_shape(self, input_shape):
30          return input_shape
31
32      def get_config(self):
33          return {'c_initializer': self.c_initializer, 'b_initializer':
                self.b_initializer,
34                  'c_trainable': self.c_trainable, 'b_trainable': self.
                    b_trainable}
```

Listing 1. Radial Softmax layer source code

The activation function has four parameters. The first two are the initial values for the b and c parameters. By default these have been initialised to vectors of zeros, but the user can set these based on their preferences or needs of the model.

The last two are boolean values which define if either b or c are learnable parameters. By default both are set as True. If any of the boolean values are set as False, then the respective vector will be set as fixed.

3 Methods

In this section we will describe the architectures, datasets and experimental setup as well as the metrics we will report. We are conducting two distinct experiments for which we report and analyse different metrics. One experiment class is done on the two moons datasets and the other on the real-life datasets.

The datasets, augmentations done and how they were collected are described in Section 3.1. We use two moons datasets, a two moons dataset and an eight moons dataset. For real-life datasets we use a total of eight of which four are used to train models with and four only for OOD testing.

In Section 3.2 we will describe the model architectures used for the different datasets. Three different types of architecture are described, basic ReLU networks, LeNets, which are convolutional neural networks (CNN) and ResNets, which are residual neural networks [12] [13] [14].

How experiments are set up is described in Section 3.3. For the moons models more visual results are given and analysed, in order to give a good understanding of how radial softmax affects the model and how it for points in the two dimensional space surrounding the trained dataset. For real-life datasets we report aggregations to know how the new activation function behaves for higher dimensional datasets.

3.1 Datasets

We class the datasets we use into two particular categories. The first two datasets are based on the two moons dataset. These datasets are used to visually show the effects of softmax and radial softmax activation functions on identical ReLU networks in a two-dimensional area around the in-distribution dataset. The two moons and eight moons datasets are described and visualised in Section 3.1. Scikit-learn was used to generate the base case for these datasets [15].

In the other category we have the real-life datasets. We use MNIST, SVHN, CIFAR-10 and CIFAR-100 for in-distribution datasets that the models are trained on [16] [**?**] [17]. These datasets as well as the augmentations done are described in Sections 3.1 - 3.1.

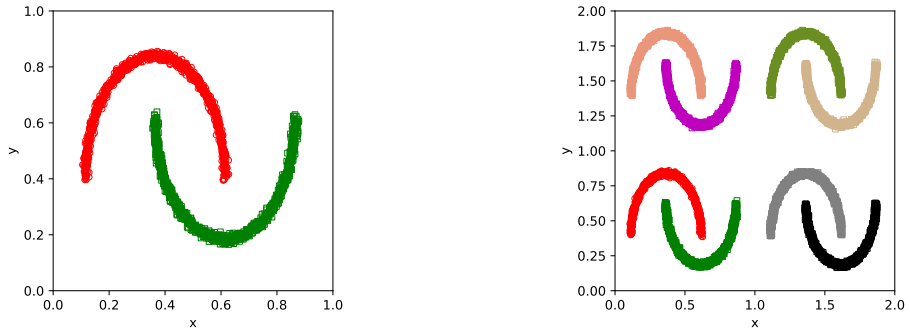
In this second category each model is tested on four OOD datasets. These datasets are FashionMNIST, EMNIST, CIFAR-10 Grayscale and LSUN Classroom [18] [19] [20]. For these datasets no augmentations other than normalization are done. The datasets are described in Sections 3.1 - 3.1.

The data pipeline from loading the data into the system, doing the necessary augmentations and feeding the data to the models during both training and testing is built using Torchvision [21].

Moons

We generated this dataset from shuffling 2000 training samples and 400 testing samples, with Gaussian noise with a standard deviation of 0.02 added. A visualisation of the dataset is in Figure 2a. We use the scikit-learn two moons generator to create the dataset [15].

Using the two moons dataset generator function we create the eight moons dataset with eight classes, seen in Figure 2b. The dataset consists of 8000 training samples and 1600 testing samples, with the two moon pairs arranged in a 2x2 grid.



(a) Two moons dataset

(b) Eight moons dataset with eight classes

Figure 2. Two moons and eight moons datasets visualised. The two moons dataset is centered around coordinates (0.5,0.5) and has two classes with 2000 data points. The eight moons dataset has eight classes and is centered around coordinates (1.0, 1.0) with 8000 data points.

The moons datasets are in a two dimensional space, where each data point has a x- and y-coordinate. The two moons dataset is centered around the point (0.5, 0.5) and the eight moons dataset is centered around (1.0, 1.0). We use eight moons as an intermediate step before more complex dataset.

MNIST

The MNIST dataset consists of 28x28 grayscale images of handwritten digits ranging from zero to nine. It consists of 60 000 training images written by 250 writers and 10 000 testing images written by 250 writers [12]. An example of the MNIST dataset can be seen at Figure 3.

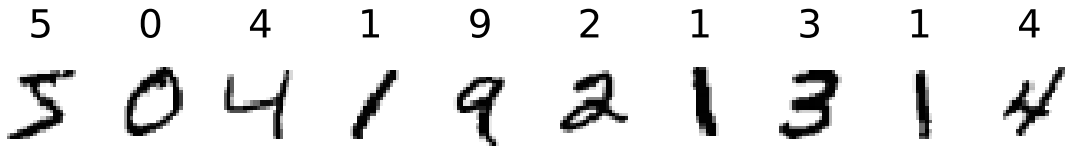


Figure 3. MNIST dataset with class labels

While training the dataset is shuffled and augmented during each epoch. The training data is normalised with a value of 255.0 and randomly cropped with a padding of 4 on each side. The random cropping step will leave the image dimensions at 28x28, however part of the original image is removed and replaced by white areas.

SVHN

The SVHN dataset, collected by Netzer, Y. et al. [**?**], consists of 32x32 coloured real-world images of house numbers. Each image is one number with some distracting digits to either side of the correct digit, which comes from the way the images are cropped. There are a total of 73 257 training images and 26 032 testing images used, with a total of 10 classes. The class labels go from 1 to 9 for those respective digits, while lass label 0 represents the number 10. For an example of the dataset, look at Figure 4. As you can see from the example, the digits on some images are easier to make out than others, this difficulty carries over to the model.

Figure 4. SVHN dataset with class labels

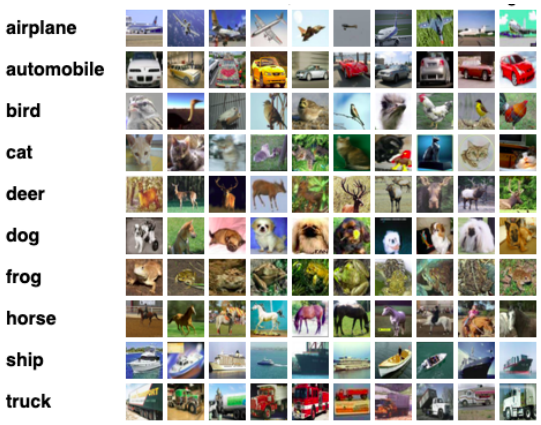
The data augmentation is the same as it was for MNIST in Section 3.1. Normalisation by dividing by 255.0 and then followed by a random crop.

CIFAR-10 & CIFAR-100

Based on the work of Krizhevsky, A. et al. [22], the CIFAR-10 dataset consists of 32x32 coloured images in 10 classes, with 6000 images in each class with no overlap between the classes. The dataset has a total of 50 000 training and 10 000 testing images. An

example of all the classes with 10 random images from each class can be seen on Figure 5a.

The CIFAR-100 dataset is also based on the work of Krizhevsky, A. et al. [22] and is similar to the CIFAR-10 dataset in the amount and size of the coloured images. However it has 100 classes which are grouped into 20 superclasses with each image coming with two labels. The first label shows the class, while the second label represents the superclass. The class labels can be seen in Figure 5b.



(a) CIFAR10 dataset [17]

Superclass	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

(b) CIFAR100 dataset [17]

Figure 5. Ten examples from each of the ten classes from the CIFAR-10 dataset on Figure 5a. Figure 5b shows the classes and superclasses of the CIFAR-100 dataset.

During the model training phase, each image is normalized by dividing each pixel by 255.0. We follow by doing a random horizontal flip with a probability of 50% and then do a random crop with a padding value of 4. After the augmentation, the images will have the same dimensionality as they did before.

EMNIST

The EMNIST dataset is based on the work of Cohen, G. et al. [19] and contains 28x28 grayscale images of handwritten characters. The dataset is derived from the NIST Special Database [16]. We use the EMNIST Letters split, which has 145 600 characters in 26 balanced classes. Classes are numeric where each class represents a letter. We use 60 000 training images and 10 000 testing images. An example of 10 letters with their respective labels can be seen on Figure 6.

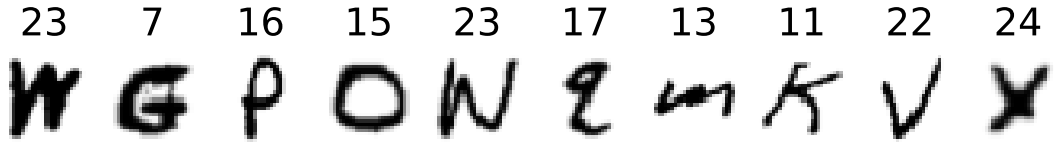


Figure 6. EMNIST dataset

The augmentations are identical to the ones used on the MNIST dataset. The training data is normalised with a value of 255.0 and randomly cropped with a padding of 4 on each side. The random cropping step will leave the image dimensions at 28x28, however part of the original image is removed and replaced by white areas.

FashionMNIST

The FashionMNIST dataset is based on the work of Xiao, H. et al. [18]. It consists of 28x28 sized grayscale images with a total of 70 000 images in 10 classes with 7 000 images per class. It is split into a training set of 60 000 images and a testing set of 10 000 images. The objects depicted are fashion items such as shoes, pants, dresses etc. For an example of the dataset where each class takes up three rows can be seen at Figure 7.

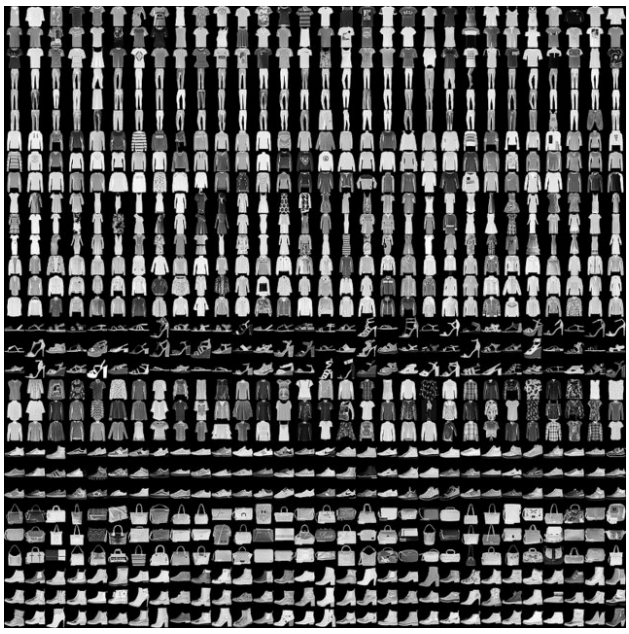


Figure 7. FMNIST dataset each class spread over three rows [18]

The augmentations are identical to the ones used on the MNIST dataset. The training data is normalised with a value of 255.0 and randomly cropped with a padding of 4 on

each side. The random cropping step will leave the image dimensions at 28x28, however part of the original image is removed and replaced by white areas.

CIFAR-10 Grayscale

The CIFAR-10 Grayscale dataset is based on the CIFAR-10 dataset collected by Krizhevsky, A. et al [22]. For this dataset we have converted the RGB colour scheme to grayscale. In Figure 8 we have visualised ten images with their respective class labels.

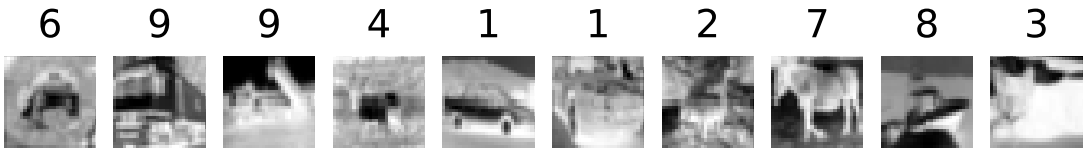


Figure 8. CIFAR-10 Grayscale dataset

We apply the same augmentation as we do for CIFAR-10. During the model training phase, each image is normalized by dividing each pixel by 255.0. We follow by doing a random horizontal flip with a probability of 50% and then do a random crop with a padding value of 4. After the augmentation, the images will have the same dimensionality as they did before.

LSUN

The LSUN dataset consists of 10 scene categories and 20 object categories, where each category is made up of one million labelled images of which a subset is labelled by humans and the rest with trained classifiers [20]. The pipeline on how this dataset was gathered is visualised in Figure 9.

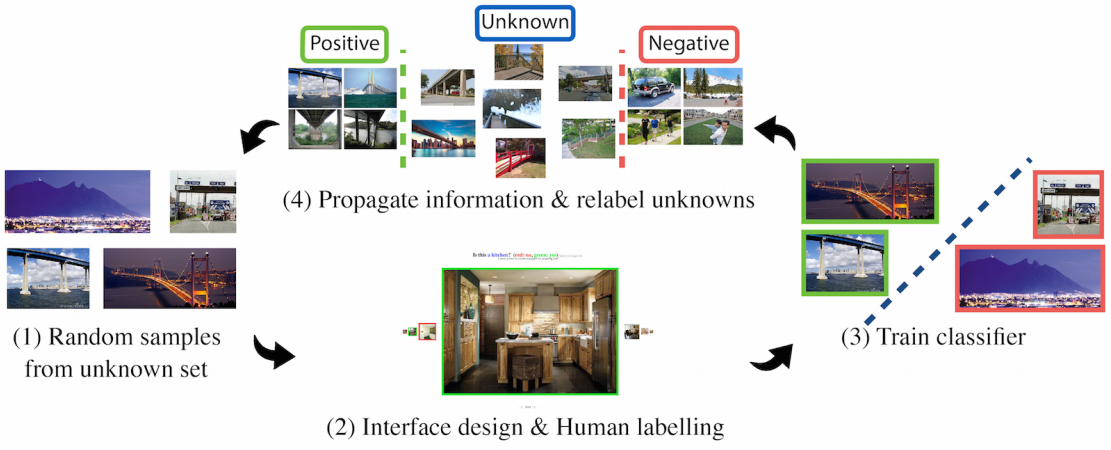


Figure 9. LSUN data gathering pipeline [20]

We used the classroom category with a split of 168 103 training and 300 testing coloured RGB images. For an example of images in the dataset, see Figure 10.

Figure 10. Example of images in the LSUN classroom category

We normalise the images by dividing each pixel by 255.0, followed by resizing the images to 32x32. The same transformations have been done on the images in Figure 10. The final dimensionality of the images is 32x32x3.

3.2 Model Architectures

Modern neural networks use the backpropagation algorithm to minimise the cost function by adjusting the networks weights and biases at each node after each forward and backward pass [23]. In this work we use four different architectures depending on the

datasets used. We have implemented each of the architectures using the Keras functional API based on the Tensorflow 2.0 backend [9, 11].

We will describe the architectures used for the moons datasets and what parameters were used while training. This is followed descriptions of the LeNet and Small ResNet architectures. The LeNet model is trained on the MNIST dataset and the Small ResNet models are trained on SVHN, CIFAR-10 and CIFAR-100 datasets [6, 13]. Each of the models is run for 100 epochs each with a batch size of 128.

Moons

For the moons datasets we will be using a basic architecture. It consists of two 100-node fully-connected layers with ReLU activation functions for the hidden layers. The output layer will be a 2-node fully-connected layer followed by either a softmax or radial softmax layer. A model with this architecture is trained on the two moons dataset for 400 epochs so that the models would converge on their optimal performance.

A model that is run on the eight moons dataset will be the same architecture as described above, instead of the 2-node output layer it will have an 8-node output layer. It will also be run for 800 epochs instead of the 400 for the two moons dataset.

The difference in epochs is to ensure the models have converged and have had sufficient training to reach their optimal performance. Both of the datasets are described in Section 3.1.

LeNet

Before describing the architecture we explain what convolutional networks are. Convolutional Neural Networks (CNN) operate with the use of convolutional layers. These layers learn image features while preserving the spatial relationship between pixels in the image. They operate by sliding a small filter, say a 5x5 pixel filter over every part of the image [12]. CNNs have been very successful in image related tasks such as image classification, recognition and captioning [24, 25, 26].

The LeNet architecture a convolutional architecture and is based on the work of LeCun, Y. et al [12]. It consists of two 5x5 convolutional layers with filters of size 32 and 64 respectively. Each convolutional layer is followed by a 2x2 max-pooling layer. on top of the convolutional layers there is a fully connected layer of size 1024 [13]. Followed by an output layer with 10 nodes with either a softmax or radial softmax activation depending on the experiment. The architecture is visualised in Figure 11.

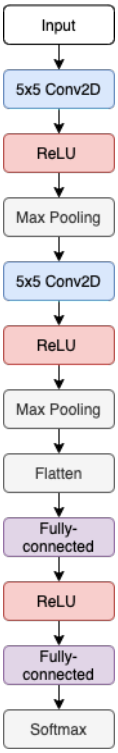


Figure 11. LeNet architecture

For the technical implementation of the architecture we followed Hein, M. et al. They use L2-regularisation on both the layer weights and biases [4]. It is shown that this architecture by itself is not resistant to adversarial or out-of-distribution attacks [13].

It is used as an experimentation model on the MNIST dataset, which is described in Section 3.1.

Small ResNet

The deeper deep neural networks become, the harder the learning becomes, because the learning signal gets weaker in the early layers (i.e. the layers far from the output). This is known as the vanishing gradient problem [14]. This is alleviated by a new architectural construct, called the residual block, as can be seen on Figure 12.

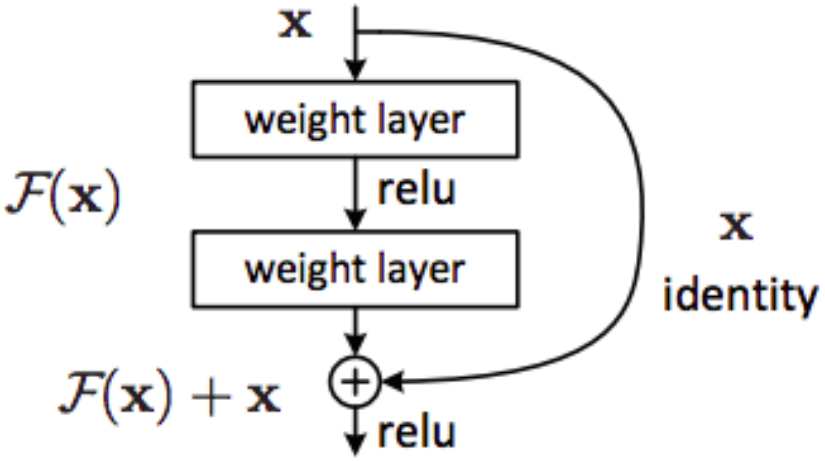
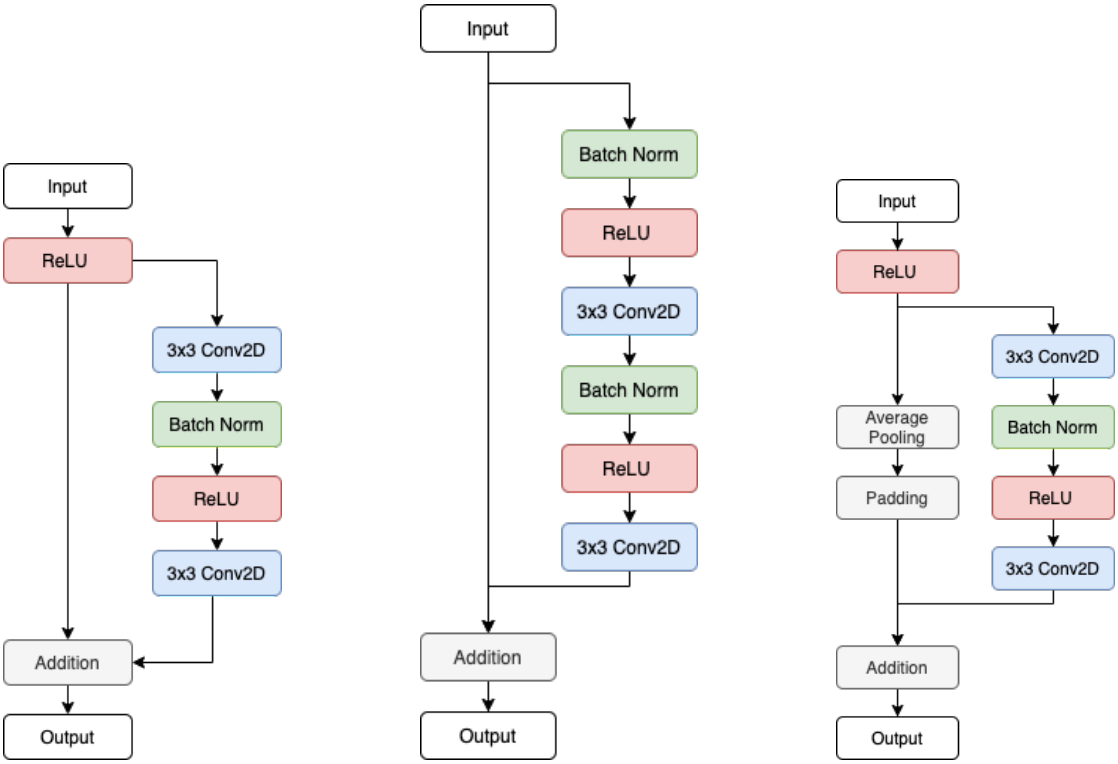


Figure 12. Residual block [14]

The identity mapping, also called a "Skip Connection", adds outputs of previous layers to the outputs of the stacked layers [27]. Since convolutional layers usually shrink their output, the identity mapping is multiplied by a linear mapping in order for the dimensions to match.

The Small ResNet model is a 19 layer ResNet architecture is based on the work of Madry, A. et al [13]. It makes use of three residual blocks. We have called these blocks A, B and C. They are visualised respectively in Figures 13a, 13b and 13c.



(a) Residual block A (b) Residual block B (c) Residual block C

Figure 13. Three residual blocks used in the Small ResNet architecture.

The three residual blocks are used to construct a residual network architecture consisting of 19 convolutional layers, which are followed by a fully-connected 10 node output layer before either a softmax or radial softmax layer. Residual block B, also called "full pre-activation", has shown to give good results in reducing classification error on CIFAR-10 [27]. The architecture for the Resnet model can be seen on Figure 14

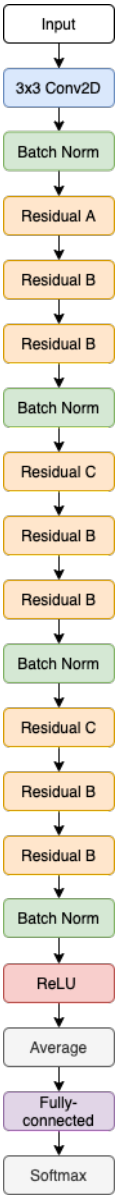


Figure 14. Small ResNet model architecture.

The most widely used residual block is block B, with either A or C being used before the B blocks. A and C blocks have to be used before the B blocks because they start with the ReLU activation layers, which follows the preceding batch normalisation layers. C blocks are used because they have the average pooling and padding layers in the skip connection, which are needed to ensure images have the correct dimensionality.

The architecture is used in models that are trained on the SVHN, CIFAR-10 and CIFAR-100 datasets, which are described in Section 3.1.

3.3 Experiment Setup

In this section we will discuss how we set up the experiments for both the moons dataset models and the real-life models. In Section 3.3.1 we describe how the models were trained and what kind of results are returned for the moons architectures, while Section 3.3.2 describes them for the real-life models. For all model runs the radial softmax parameters are left at their default values. Both maximum confidences and class balances are initialised as vectors of ones and are set as learnable.

3.3.1 Moons

For the moons datasets described in Section 3.1, we will be using the two moons and eight moons architectures described in Section 3.2. While training we will use the Adam optimiser with a learning rate of 0.01 and categorical cross-entropy as the loss function [28]. The two moons model will be trained for 400 epochs and the eight moons model for 800 epochs so that they would converge on their optimal performance.

The moons models are used to visually compare the effect that softmax and radial softmax have on the models. Specifically how the model behaves on data around the in-distribution dataset. We also plot the class scores and probabilities of the output activation layer for each class. For this we gather OOD data that surrounds the in-distribution data and data from the middle of the in-distribution dataset expanding out.

3.3.2 Real-life Datasets

For the real-life datasets we have two distinctive model architectures, LeNet and Small ResNet, which are described in Section 3.2. We use the LeNet architecture to train on the MNIST dataset [4, 13]. The Small ResNet architecture is used to train the SVHN, CIFAR-10 and CIFAR-100 models [4].

The models are trained for 100 epochs each with the Adam optimiser with a learning rate of 0.001 for MNIST and stochastic gradient descent with momentum with a learning rate of 0.1 and momentum at 0.9 for SVHN, CIFAR-10 and CIFAR-100 [28, 29]. A learning rate scheduler is applied for all models which reduces the learning rate by a factor of 10 after the 50th, 75th and 90th epochs.

We test each models resilience to OOD samples by using it to score samples taken from different OOD datasets. Which OOD datasets are used for each model are brought out in Table 1. Four metrics are used to compare the performance of softmax and radial softmax [3, 4, 30].

The metrics are defined as follows:

1. Test error - defined as $1 -$ accuracy.
2. Mean Maximum Confidence (MMC) - defined as the mean of maximum confidence values for each data point scored by the model.

3. Area Under the Receiver Operating Characteristics (AUROC) - represents a measure of separability as in how good the model is in predicting the correct class. Plotted with true positive ratio (TPR) on the y-axis and false positive ratio (FPR) on the x-axis and defined as follows:
    - TPR - defined as $\frac{TP}{TP+FN}$
    - FPR - defined as $\frac{FP}{TN+FP}$
4. False positive rate at 95% true positive rate (FPR@95) - defined as the FPR when the TPR is at 95%

In determining if radial softmax is successful in detecting OOD samples, we want the MMC for in-distribution datasets to be high and lower for OOD datasets. For AUROC we want it to be higher than it is for softmax and we are looking for the FPR@95 score to be lower. Based on the described combination we know that radial softmax is better at detecting OOD samples than softmax [30].

Table 1. Matrix noting what datasets are used to test models trained on specific datasets.

	Testing Dataset							
Trained on	MNIST	EMNIST	FashionMNIST	SVHN	CIFAR-10	CIFAR-10 Grayscale	CIFAR-100	LSUN Classroom
MNIST	X	X	X			X		
SVHN				X	X		X	X
CIFAR-10				X	X		X	X
CIFAR-100				X	X		X	X

For each model we run the experiment five times and average out the results. This is done to lower the effects of any random noise that might affect the training of the datasets. The training data is also shuffled after each epoch.

4 Main Results

In this section we will present the results gathered from our experiments as described in Section 3. This section is split into two subsections, based on if the experiments were done on the moons datasets or real-life datasets, following the description in Section 3.3.

The moons datasets models' results and analysis are brought out in Section 4.1 which has a section for both two moons and eight moons models. This subsection is very visual in order to give an intuitive understanding of how the two softmax functions affect the models in a two dimensional space.

The real-life datasets results are presented in Section 4.2, where the MMC, AUROC and FPR@95 metrics defined in Section 3.3.2 are used to compare the performance of radial softmax against softmax in regards to OOD resilience.

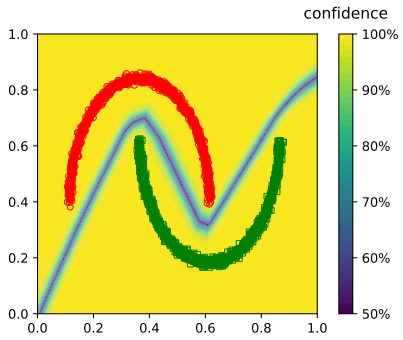
4.1 Experiments on the Moons Datasets

The moons datasets are used to get a good understanding in a two dimensional space of how radial softmax behaves differently in comparison to softmax. We have two subsections in this section, the first is for the two moons dataset in Section 4.1.1 and the second is on the eight moons dataset with eight classes in Section 4.1.2.

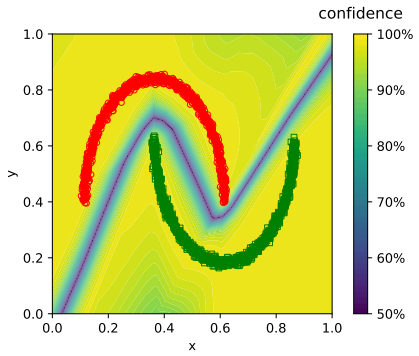
In both sections we are presenting the confidence areas, class scores and class probabilities in the two dimensional space around the training dataset. We also gather points from the middle of the training dataset in a straight line to a point OOD. With this we can see how the effects of the activation function as we gather data from progressively farther from the training dataset.

4.1.1 Two Moons

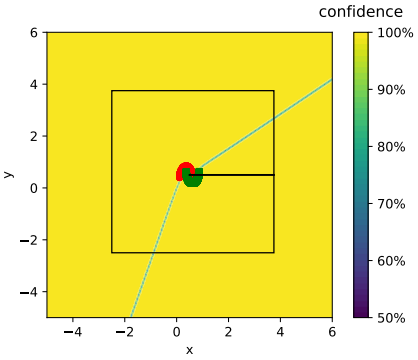
After training models with softmax and radial softmax for 400 epochs we reached a testing accuracy of 100% in both cases. This allows us to visually compare the models' behaviour without having to take into consideration if one model is more accurate than the other and if there is a trade-off between classification accuracy in the original task and in the ability to detect OOD data.



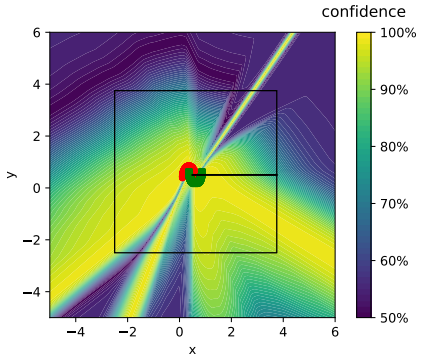
(a) Zoomed in softmax confidence area



(b) Zoomed in radial softmax confidence area



(c) Zoomed out softmax confidence area with OOD data box

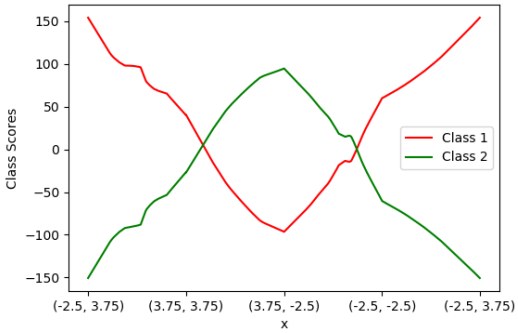


(d) Zoomed out radial softmax confidence area with OOD data box

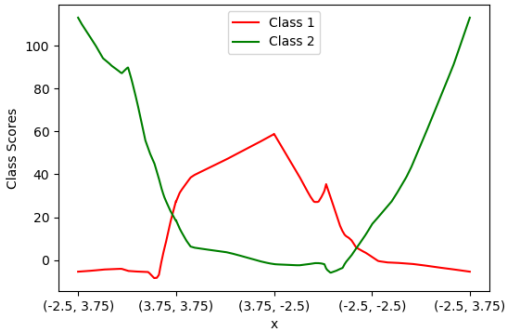
Figure 15. Visualised confidence areas for two moons models trained for 400 epochs. Figures 15a and 15c visualise the softmax model both zoomed in and zoomed out. Figures 15b and 15d visualise the same for radial softmax. The black boxes in Figures 15c and 15d mark lines where OOD data was taken from.

We can see from Figure 15a that the softmax model was not successful in neither limiting the high confidence area nor lowering the confidence scores. However there is a small separator at 50% confidence between the two moons. From Figure 15c we can see that high confidence goes on further OOD.

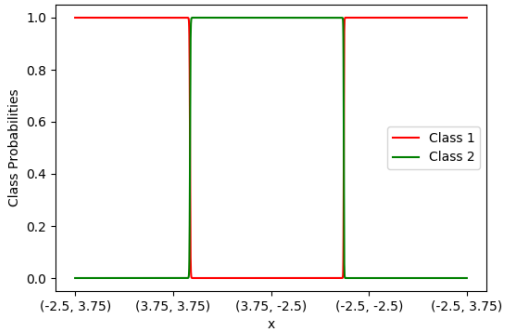
In comparison in Figure 15b even though a 50% confidence area was not created in all directions around the moons, the confidence scores were more limited. Visually evaluating it is around the 90% mark around the moons. Observing Figure 15d, radial softmax was successful in creating areas of 50% confidences and at gradually lowering the confidence scores farther into the two-dimensional space.



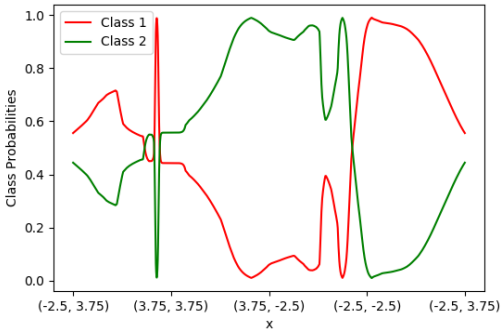
(a) Softmax function inputs



(b) Radial softmax function inputs



(c) Softmax function output class probabilities

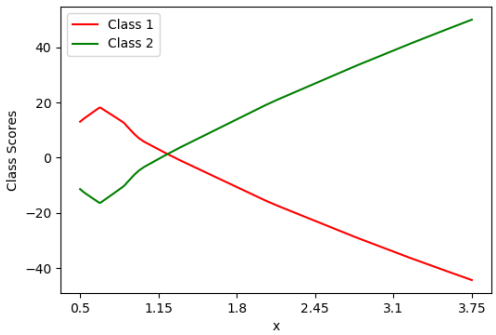


(d) Radial softmax function output class probabilities

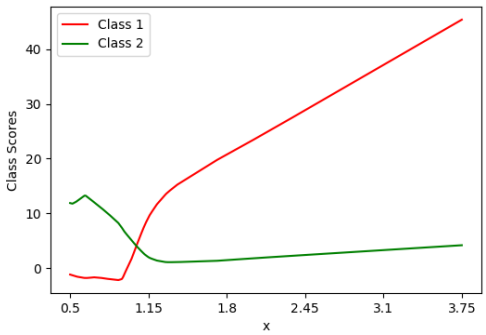
Figure 16. Visualisations of the two moons class scores in Figures 16a and 16b. Figures 16c and 16d visualise the output class probabilities. X-axis has the coordinates of the square on which edges the data points lie and the y-axis has either the class scores or class probabilities.

Comparing Figures 16a and 16c to Figures 16b and 16d, there is a distinct effect that radial softmax has had on both the class scores and class probabilities. On the softmax figures we see a mirroring effect on the y-axis for the two classes. For example on Figure 16a right after the point (3.75, 3,75) class 2 has a score of 100 and class 1 has a score of -100. This mirroring can be seen throughout the graph. As seen on Figure 16c, whenever one of the class scores is above 0, that class probability 1 and the other is 0. There are only very limited areas of probabilities between 0 and 1 on this experiment with softmax.

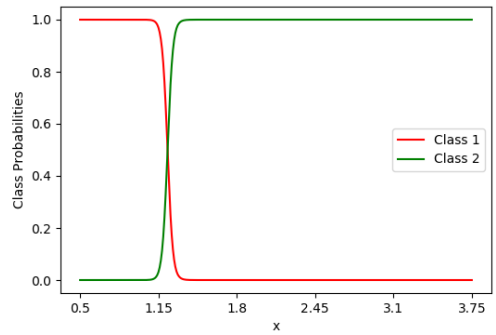
Radial softmax was successful in pushing class scores closer to 0 and there is no mirroring effect in Figure 16b. The effect of radial softmax is more prominent in Figure 16d in terms of class probabilities. Namely there is more expressiveness in class probabilities as in they are not binary, but more continuous.



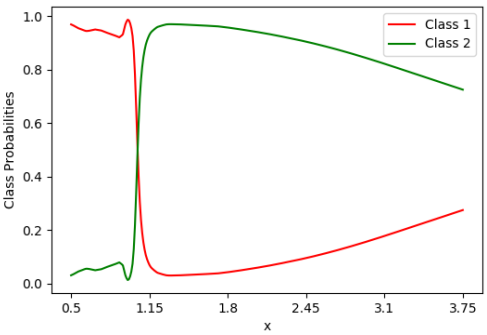
(a) Softmax inputs for data points from (0.5, 0.5) to (3.25, 0.5)



(b) Radial softmax inputs for data points from (0.5, 0.5) to (3.25, 0.5)



(c) Softmax outputs for data points from (0.5, 0.5) to (3.25, 0.5)



(d) Radial softmax outputs for data points from (0.5, 0.5) to (3.25, 0.5)

Figure 17. Class scores and probabilities for data points gathered from the middle of the dataset at (0.5, 0.5) up till (3.25, 0.5) in 0.01 x-axis steps. These are visualised in Figures 17a and 17c and Figures 17b and 17d for softmax and radial softmax models respectively.

Softmax class scores behave similarly in the outwards experiment in Figure 17a, as they did in the square experiment in Figure 16a. As much as one class score is positive, in this case class 2, the other class is negative, in this case class 1. The only point at which they are balanced is in (1.25,0.5). This behaviour is observed with the class probabilities as well. In Figure 17c, whichever class has a positive class score, mostly has 100% probability. The only exception is the region around the point (1.25,0.5), where the class scores are balanced and the probability goes 50% in a narrow area.

With the radial softmax model Figures 17b and 17d visualise how radial softmax performs as data points get progressively farther from the in distribution data. Even though the difference in class scores increases when going further OOD, radial softmax by its definition is able to counteract this. As both class scores increase, the model returns

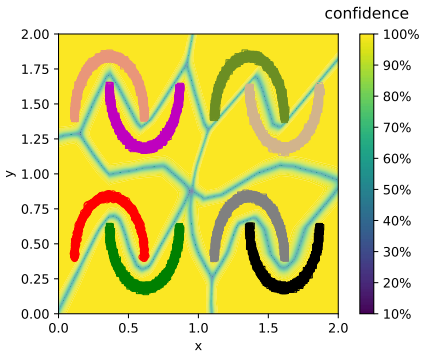
to the learned class distribution, which should be approximately equal to the actual class distribution. We can see it starting to converge towards the class distribution in Figure 17d.

In this section we saw the differences between softmax and radial softmax models in two dimensional space with the two moons dataset. Figure 15d shows that radial softmax is able to create areas of balanced class probabilities more effectively than softmax. As we moved away from the training data, radial softmax started moving the two class probabilities closer to the balance point in Figure 17d.

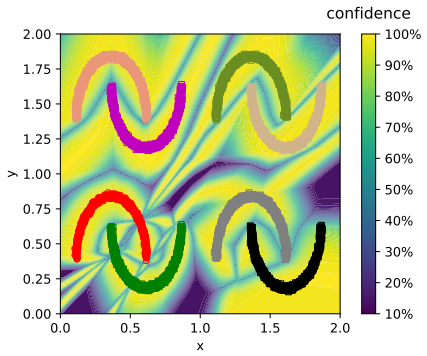
In the following subsection we will present results on how softmax and radial softmax performed in the same two-dimensional input space, with a more complex dataset.

4.1.2 Eight Moons

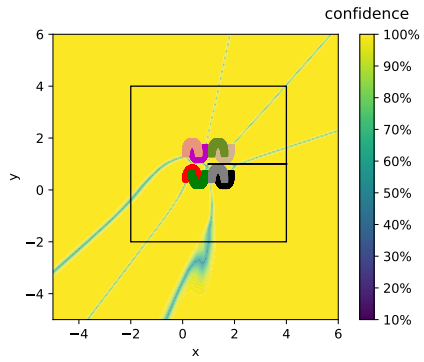
The moons model architecture with both softmax and radial softmax was trained for 800 epochs per model and both models reported a testing accuracy of 100% on a testing set of 800 data points.



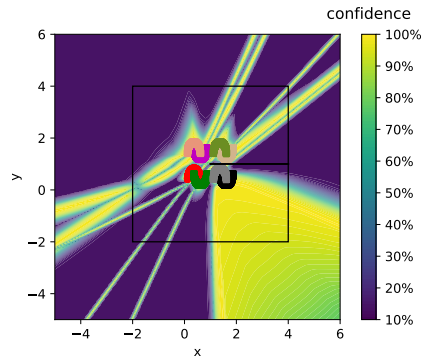
(a) Zoomed in softmax confidence area



(b) Zoomed in radial softmax confidence area



(c) Zoomed out softmax confidence area

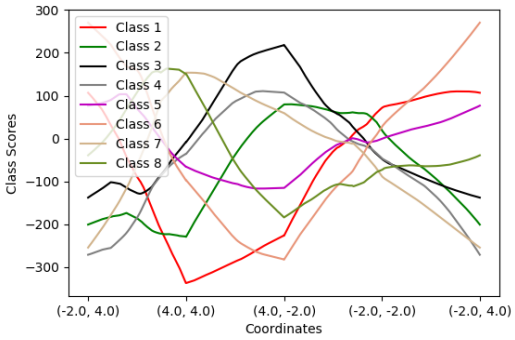


(d) Zoomed out radial softmax confidence area

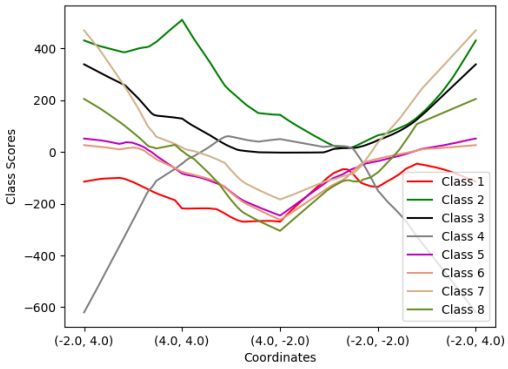
Figure 18. Visualised confidence areas for eight moons models trained for 800 epochs. Figures 15a and 18c visualise the softmax model both zoomed in and out. Figures 18b and 18d visualise the same for radial softmax. The black boxes in Figures 18c and 18d mark lines where OOD data was taken from.

The softmax model was able to separate each of the classes and in Figure 18a there are distinct closed off areas for the purple and grey classes. However the model does not predict 12.5% probability, which is the class balance with 8 classes, for the separator lines. Instead it predicts approximately 50% probability. A wider look at the two dimensional space around the training dataset in Figure 18c shows that the model was not able to limit high confidence areas, similarly to the two moons test.

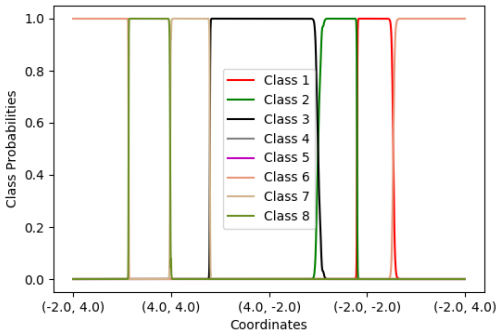
The radial softmax model was able to bring the high confidence areas closer to the training data points with eight classes, than it was for two classes, Figure 18b vs. Figure 15b. This is possibly due to there being more classes closer to each other, thus limiting each classes possible areas. The radial softmax model also had more success in creating areas of approximately balanced class probabilities further from the dataset with the eight moons dataset than it did for the two moons dataset, Figure 18d vs. Figure 15d.



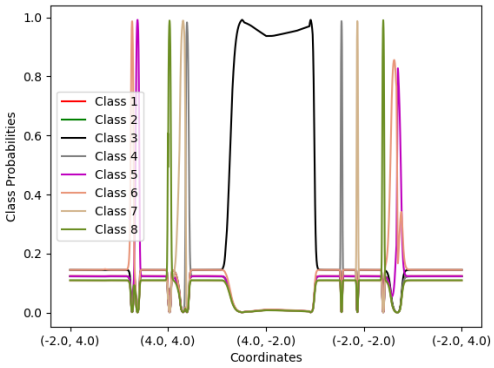
(a) Softmax function inputs eight moons



(b) Radial softmax function inputs eight moons



(c) Softmax function outputs eight moons

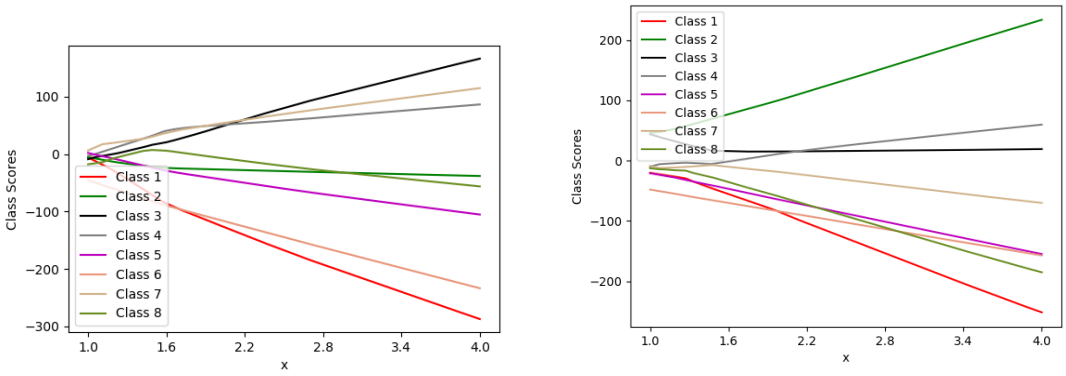


(d) Radial softmax function outputs eight moons

Figure 19. Visualisations of the eight moons class scores in Figures 19a and 19b for the softmax and radial softmax models respectively. Figures 19c and 19d visualise the class probabilities for the softmax and radial softmax models respectively. X-axis has the coordinates of the square on which edges the data points lie and the y-axis is either the class scores or class probabilities.

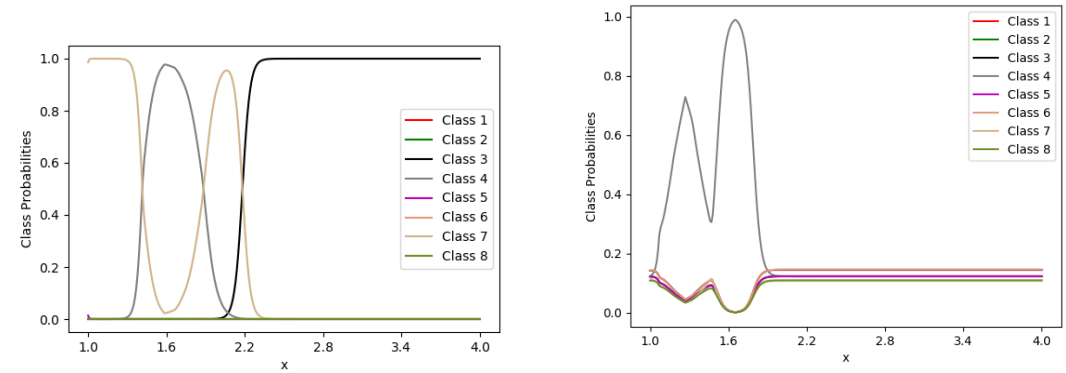
Due to the higher complexity of the training set, we do not observe the mirroring effect in Figure 19a. However the maximum and minimum values for class scores have increased, 300 and over -300 in comparison to 150 and -150 for the two moons in Figure 16a. The class probabilities in Figure 19c are similarly binary and lack expressiveness.

The class probabilities in Figure 19d are mostly balanced at approximately 12.5%, which is our uniform class distribution. However, in some zones we see also high confidence up to near 100% (e.g. the area corresponding to bottom right corner of the square). This means that at those OOD areas, radial softmax did not achieve what it was designed for.



(a) Softmax inputs for data points from (1.0, 1.0) to (4.75, 1.0)

(b) Radial softmax inputs for data points from (1.0, 1.0) to (4.75, 1.0)



(c) Softmax outputs for data points from (1.0, 1.0) to (4.75, 1.0)

(d) Radial softmax outputs for data points from (1.0, 1.0) to (4.75, 1.0)

Figure 20. Class scores and probabilities for data points gathered from the middle of the dataset at (1.0, 1.0) up till (4.75, 1.0) in 0.01 x-axis steps. These are visualised in Figures 20a and 20c and Figures 20b and 20d for softmax and radial softmax models respectively.

The softmax model has kept most of the class scores below zero in Figure 20a, however the probabilities are still mostly binary, with class 3 dominating for most of the selected data points, Figure 20c. Before the point (2.25,1), the softmax model's class probabilities are not binary and there is some exchange between classes 4 and 7. After the point (2.25,1) class 3 has essentially 100% probability.

For radial softmax the class scores plot from Figure 20b seems very similar to softmax, since there is similar fanning of the scores and except for classes 3 and 4, all class scores become very far from 0 as we move further into OOD. Radial softmax class probabilities show very good results in Figure 20d. None of the class probabilities

report 100% probability and after the point (2,1) all classes report a class probability of approximately 12%, which is near the actual class balance. These results are very promising and support our design decisions.

From Section 4.1.1 we see that for a simple architecture and a simple two class dataset, radial softmax was able to lower both the areas of confidence and the levels as well. The results also show that as discussed in Section 2.1, softmax is unable to control confidence areas and recognise OOD data.

Using a more complex dataset with eight moons and eight different classes in Section 4.1.2 we can see that radial softmax performed better on the more complex dataset, than it did on two moons.

In the following section we will look at how radial softmax is able to perform with real-life datasets.

4.2 Real-life Datasets

In Table 2 we have reported the averaged results over five models on the MNIST dataset with LeNet and on the SVHN, CIFAR-10 and CIFAR-100 datasets with the Small ResNet. The results where radial softmax was better in comparison to regular softmax are marked out in bold. For the radial softmax to be considered as having better results, MMC must be smaller for OOD datasets, but higher or equivalent for the datasets it was trained on. AUROC has to be higher and FPR@95 must be lower. Both TE and TL must also be equivalent or lower.

Table 2. Average results over five model runs on the MNIST dataset with LeNet, SVHN, CIFAR-10 and CIFAR-100 with ResNet respectively. The metrics that were better with radial softmax are marked in bold.

Trained on	Softmax (TE: 0.56%, TL: 0.068)			Radial Softmax (TE: **0.55%**, TL: 0.071)		
MNIST	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
MNIST	0.99	-	-	0.989	-	-
EMNIST	0.818	0.889	0.351	0.82	**0.89**	0.358
FMNIST	0.69	0.968	0.142	**0.68**	**0.975**	**0.13**
CIFAR-10Grayscale	0.494	0.995	0.006	**0.492**	0.995	**0.001**
Trained on	Softmax (TE: 3.57%, TL: 0.23)			Radial Softmax (TE: **3.48%**, **TL: 0.21**)		
SVHN	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
SVHN	0.978	-	-	0.975	-	-
CIFAR-10	0.722	0.939	0.339	**0.665**	**0.945**	**0.278**
CIFAR-100	0.723	0.935	0.342	**0.664**	**0.943**	**0.281**
LSUNClassroom	0.728	0.938	0.348	**0.663**	**0.951**	**0.271**
Trained on	Softmax (TE: 8.48%, TL: 0.42)			Radial Softmax (TE: 8.77%, TL: 0.43)		
CIFAR-10	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
CIFAR-10	0.949	-	-	0.941	-	-
SVHN	0.743	0.887	0.683	0.75	0.869	0.732
CIFAR-100	0.767	0.885	0.712	**0.746**	0.845	**0.709**
LSUNClassroom	0.735	0.878	0.674	**0.687**	**0.888**	**0.61**
Trained on	Softmax (TE: 31.89%, TL: 1.52)			Radial Softmax (TE: 32.67%, TL: 1.54)		
CIFAR-100	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
CIFAR-100	0.748	-	-	0.701	-	-
SVHN	0.571	0.701	0.837	**0.455**	**0.755**	**0.826**
CIFAR-10	0.562	0.715	0.852	**0.5**	0.713	0.854
LSUNClassroom	0.549	0.725	0.849	**0.477**	**0.734**	**0.848**

First looking at the TE, TL and MMC performances on the trained datasets, radial softmax has better TE performance on two datasets, namely MNIST and SVHN, with a better TL value with SVHN. MNIST is also the simplest of the datasets. Looking at the training data MMC, radial softmax underperformed in comparison to softmax on all models. However since radial softmax is defined with the intent of minimising model confidence for OOD data, it might also have the side effect of lowering the confidence for in-distribution data.

Continuing to the out-of-distribution datasets for MNIST, radial softmax outperformed softmax in six out of nine results. In all cases the differences between softmax and radial softmax are within the range of +/-0.1. We cannot therefore claim that radial softmax would have increased the OOD dectection ability of the LeNet model trained on MNIST.

Analysing the models run on SVHN, radial softmax outperformed softmax in all of the metrics for OOD datasets. The average difference for MMC and FPR@95 is at 0.06. For AUROC it is at 0.009. Differences for all metrics are better than for MNIST. The

largest differences in terms of results are for the LSUNClassroom dataset, which is also the largest difference out of all OOD datasets including the ones used for MNIST. SVHN is more complex than MNIST, however it is still quite different visually from the OOD datasets that we tested the model on.

The CIFAR-10-trained radial softmax model performed similary to the MNIST model in terms that five out of nine results were better with radial softmax, than they are with softmax. The average differences for MMC and AUROC are at 0.017 and 0.016 respectively, where for AUROC softmax performed better. For FPR@95 it was much lower, at 0.006. Again in average the largest differences in respect to the other OOD datasets is LSUNClassroom.

The CIFAR-100 model performed better than the CIFAR-10 and MNIST models in terms that only two out of nine results were better with softmax, in comparison to three out of nine for MNIST and CIFAR-10. On average MMC for radial softmax was better by 0.08, with the highest difference being for the SVHN dataset, where it was 0.116 and the smallest was for CIFAR-10, where it was at 0.062. The average difference for AUROC was at 0.02, where the smallest was for CIFAR-10 at 0.002. For FPR@95 the average difference was 0.003. Radial softmax was the most effective in reducing the MMC for OOD data. However for the other metrics it was not as effective in improving performance.

Radial softmax had success in limiting both the confidence areas and levels of confidence on the two moons dataset in comparison to the same model with softmax in Figures 15c vs Figure 15d for softmax and radial softmax models respectively. The proposed function also was more successful on the eight moons dataset, where it brought the low confidence close to the training data, as seen in Figure 18d. It also acted as designed in predicting class probabilities far from the training data at the learned class balance, which approximated the actual class balance, as seen on Figure 20d. With the moons datasets the new function showed good results.

However, with the real-life datasets radial softmax did not offer much improvement in comparison to regular softmax in detecting OOD data points, as seen in Table 2. However none of the reported metrics also did show signs of being much worse in comparison to regular softmax. The worse comparative performance on the real-life datasets might be because both the data and models used are much more complex than the ones used on the moons datasets. This makes it possible that the high confidence behaviour is learned in the numerous hidden ReLU layers and radial softmax as the output activation function is not able to have such a large effect on the very deep architectures.

5 Ablation Study

For the ablation study we will train three models on an eight moons two classes dataset described in Section 5.1. In this section we will demonstrate that it is possible for radial softmax to learn incorrect parameters and how fixed parameters can be used instead. The results are presented in Section 5.2.

5.1 Experiment Setup

For this experiment we use the same model architecture that was used for the eight moons dataset, described in Section 3.2. We train two models for 800 epochs to have comparable results to the eight moons results in Section 4.1.2. One of the models uses softmax and the other radial softmax. The third model uses softmax with fixed b and c parameters and is trained for 1200 epochs.

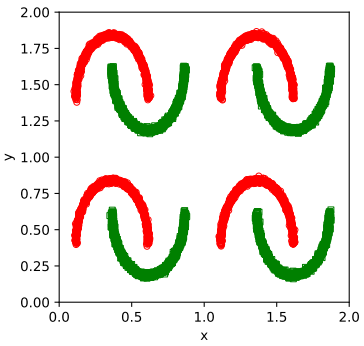


Figure 21. Eight moons dataset with two classes in a two dimensional space.

In Figure 21 we have visualised the eight moons dataset with two classes. Similarly to the regular eight moons dataset it is centered around the point (1,1) and consists of a total of 8000 training samples and 1600 testing samples.

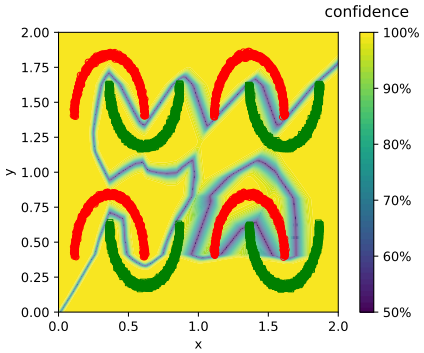
5.2 Results

We demonstrate in Section 5.2.1 that the radial softmax can learn the wrong b and c parameters for a relatively simple dataset and provide an analysis based on the b and c parameters why it failed to limit areas of confidence without reduction in performance for in-distribution data.

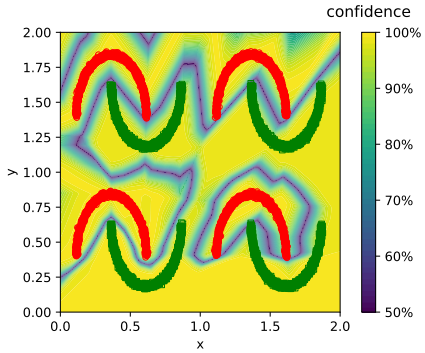
In Section 5.2.2 we demonstrate that by fixing the b and c parameters radial softmax is able to act as designed on datasets that it failed to learn the correct parameters for. This demonstrates use cases for parameter tuning.

5.2.1 Learnable Class Distribution and Maximum Probability

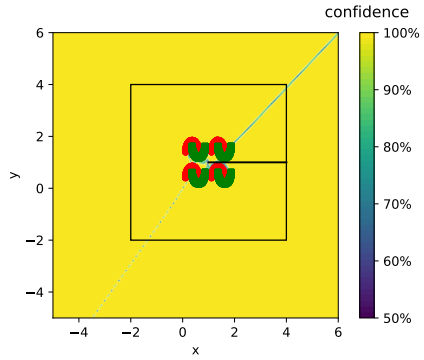
We trained both the softmax and radial softmax models on the training set for 800 epochs and then tested on the testing set. We were able to get a testing accuracy of 100% for both of the models. We report the same graphs as we did in Section 4.1 and the learned b and c values of both the two moons and eight moons two classes models.



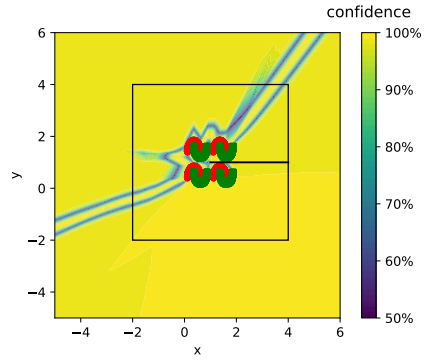
(a) Zoomed in softmax confidence area



(b) Zoomed in radial softmax confidence area



(c) Zoomed out softmax confidence area



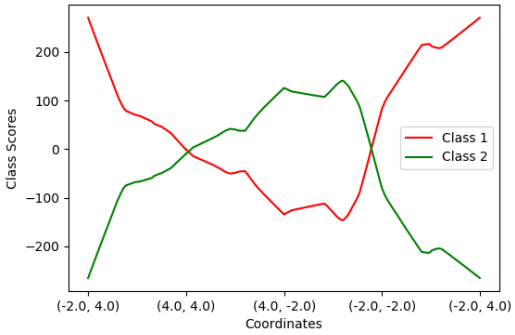
(d) Zoomed out radial softmax confidence area

Figure 22. Visualised confidence areas for eight moons two classes models trained for 800 epochs. Figures 22a and 22c visualise the softmax model both zoomed in and out. Figures 22b and 22d visualise the same for radial softmax. The black boxes in Figures 22c and 22d mark lines where OOD data was taken from.

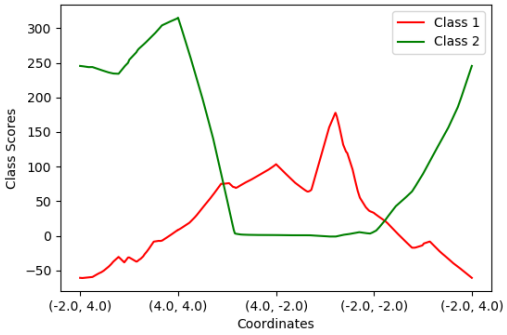
Looking at the softmax confidence areas on Figures 22a and 22c, we can see that it has performed similarly as it did in Sections 4.1.1 and 4.1.2. It was unable to limit the confidence area or lower confidence.

In this experiment radial softmax was unable to limit the areas of confidence, like it was for either two moons or eight moons in Sections 4.1.1 and 4.1.2 respectively. It was more effective in lowering confidence areas around the moons in comparison to softmax,

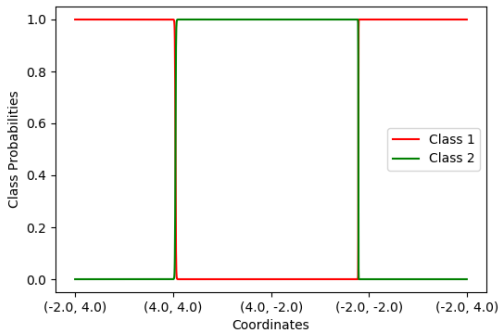
Figures 22a vs. 22b. A lower confidence level is also for around 75% of the area around the moons in Figure 22d, however confidence is reduced only slightly to around 95%. These results are underperforming in comparison to the two moons or eight moons in Figures 15d and 18d respectively.



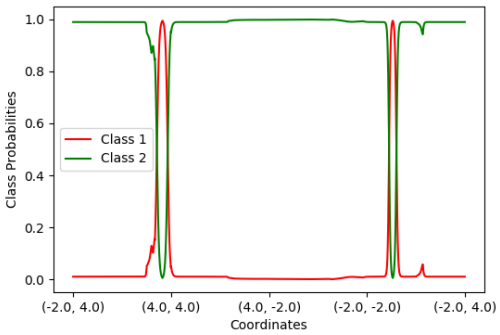
(a) Softmax function class scores eight moons



(b) Radial softmax function class scores eight moons



(c) Softmax function class probabilities eight moons



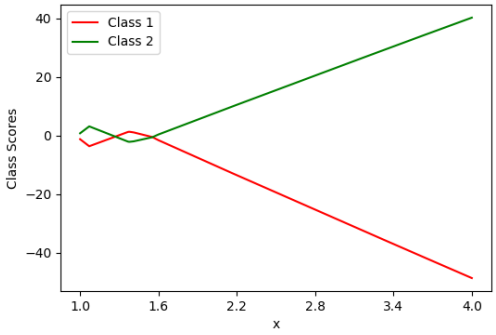
(d) Radial softmax function class probabilities eight moons

Figure 23. Visualisations of the eight moons two classes softmax class scores in Figures 23a and 23b. Figures 23c and 23d visualise the class probabilities. X-axis has the coordinates of the corners of the square on which edges the data points lie and the y-axis is either the function inputs or outputs.

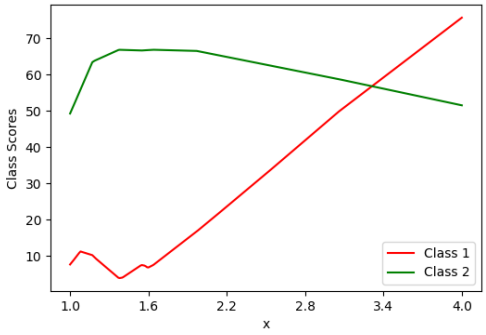
This softmax model has performed similarly to the softmax model trained on two moons in Section 4.1.1. The class scores are mirrored of each other in Figure 23a and the class probabilities are binary, either 100% or 0% in Figure 23c. The scores do have larger maximum and minimum values in comparison to the two moons model.

With the radial softmax model in Figures 23b and 23d the mirroring effect was removed for the class scores and they are pushed closer to 0. However the probabilities

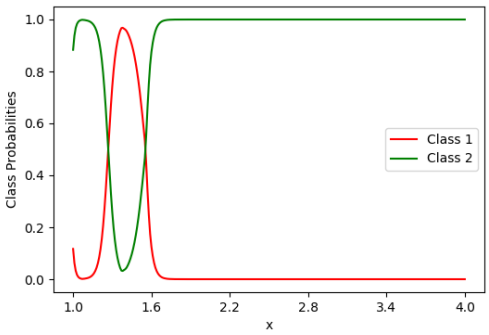
are nearly binary and they do not follow our intended design of radial softmax by which values closer to 0 should lead to higher class probabilities. The possible cause of this effect is explored further in this section, where we also analyse the learned b and c parameters for the model.



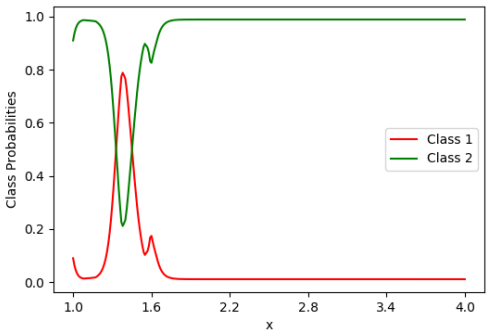
(a) Softmax inputs for data points from (1.0, 1.0) to (4.75, 1.0)



(b) Radial softmax inputs for data points from (1.0, 1.0) to (4.75, 1.0)



(c) Softmax outputs for data points from (1.0, 1.0) to (4.75, 1.0)



(d) Radial softmax outputs for data points from (1.0, 1.0) to (4.75, 1.0)

Figure 24. Class scores and probabilities for data points gathered from the middle of the dataset at (1.0, 1.0) up till (4.75, 1.0) in 0.01 x-axis steps. These are visualised in Figures 24a and 24c and Figures 24b and 24d for softmax and radial softmax eight moons two classes models respectively.

The softmax model behaved similarly in Figures 24a and 24c as it did on the two moons dataset in Section 4.1.1. The class scores have a similar wedge shape, however the scores are closer to 0 for a longer period of time. The class probabilities are mostly binary except for a period before the point (1.75, 1), during which it changes predictions.

The radial softmax model in Figures 24b and 24d has not performed as designed. First for the class scores it was unable to push the scores close to 0, however class 1 does

start out near it and class 2 score is decreasing. Interestingly the class probabilities in Figure 24d are stably binary and even after the class score for class 1 becomes dominant, the class probabilities stay the same. This effect is also analysed more below.

Table 3. B and c parameter values for classes after radial softmax models trained on two moons and eight moons two classes datasets.

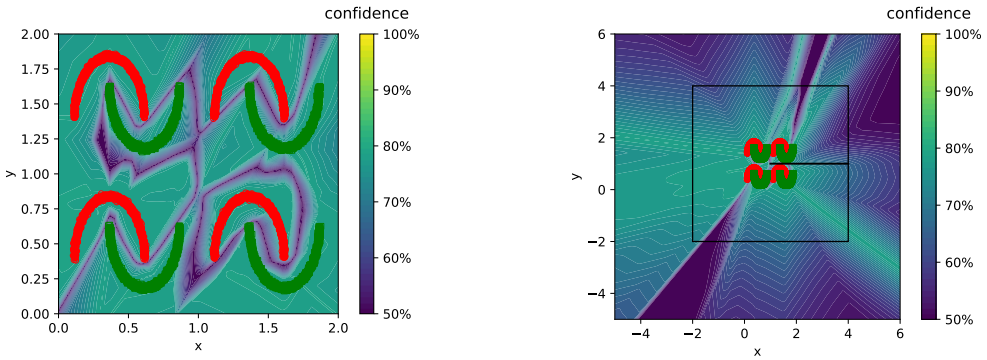
	b		c	
Trained dataset	class 1	class 2	class 1	class 2
Two moons two classes	-2.091	-1.88	2.123	1.874
Eight moons two classes	-4.957	-0.414	4.835	2.142

In Table 3 we can see the learned b and c parameters for both of the classes after training models on the two moons and eight moons two classes datasets. Since the classes in both datasets are balanced and we use full batch training, the b and c values for the two classes should be approximately equal. We can see that this is so for the successful run on the two moons dataset. Looking at the eight moons two classes run, we can see that the model was not successful in learning neither the b or c parameters correctly.

Let us take two points from Figure 24b. We will take one before the class scores cross, where $z_1 = 30$ and $z_2 = 60$. For the other point we will take it after the class scores have crossed, where $z'_1 = 70$ and $z'_2 = 50$. Taking into consideration the definition of radial softmax in Equation 3, we can see that the z-values are so far from 0, that they overpower the c values and as such radial softmax should return the class distribution defined by b-s. However the b values have been learned in a way that the class 2 probability dominates over the class 1 probability.

5.2.2 Fixed Class Distribution and Maximum Probabilities

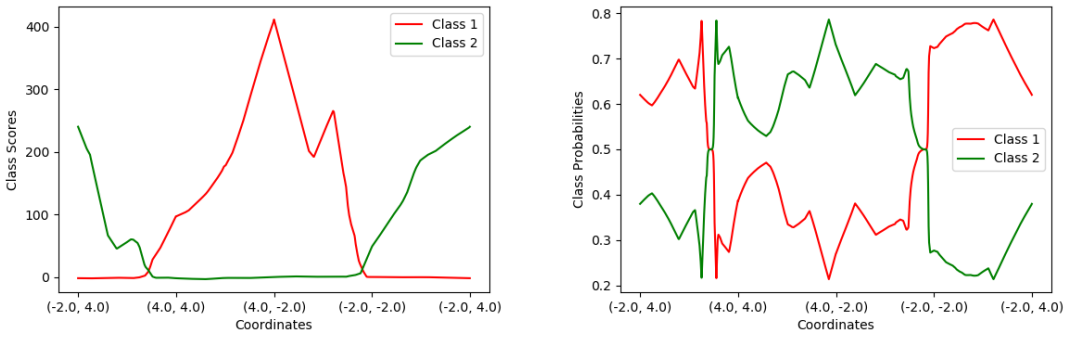
We trained a radial softmax model with fixed parameter values. For b the values were 0.01 for both of the classes and for c the values were both 1. These values give out a numerator range, which is expressive and should successfully predict near-balance for OOD data points. The model was trained for 1200 epochs to get 100% testing accuracy.



(a) Zoomed in radial softmax confidence area (b) Zoomed out radial softmax confidence area

Figure 25. Confidence areas for a radial softmax model trained on an eight moons two classes dataset with fixed b and c parameters. Black lines represent areas where OOD data points were drawn from.

Comparing to the results in Section 5.2, where the model was able to learn the b and c parameters, it had better success with fixed parameters and more training epochs. The model is able to have reasonable success in detecting OOD samples from near the moons in Figure 25a. From the zoomed out view in Figure 25b, we can see areas of balanced confidence and that the higher confidence areas gradually fade to lower confidence areas.

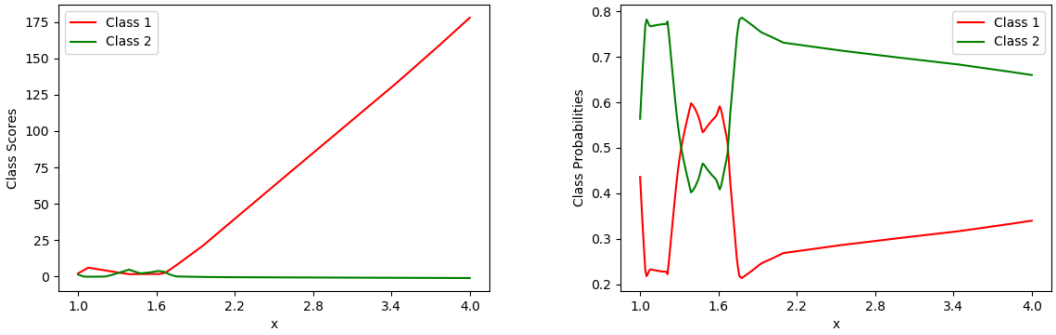


(a) Fixed parameter radial softmax class scores (b) Fixed parameter radial softmax class probabilities

Figure 26. Class scores and probabilities for OOD samples taken from a square area from around the training dataset of a fixed parameter radial softmax model.

The radial softmax model was successful in pushing the class scores down to nearly 0 in Figure 26a and whenever either of the classes were not around 0, it had a very large class score. The radial softmax function was not successful in predicting a balanced

probability for most of the samples as can be seen in Figure 26b. It was successful in bringing down the maximum probabilities, which also came at the cost of increasing the minimum probabilities, which were 80% and 20% respectively. This limiting of the probabilities comes from how the class scores interact with the parameters.



(a) Fixed parameter radial softmax class scores (b) Fixed parameter radial softmax class probabilities

Figure 27. Class scores and probabilities for OOD samples generated following a line from the middle of the training dataset at point (1.0,1.0), till the point (4.0,1.0), predicted by the fixed radial softmax model.

The class scores on Figure 27a indicate that as we get farther from the in-distribution dataset, the class score of class 1 increases linearly. This is to be expected as from Figure 25a we can see that both of the red half-moons to the right side of the dataset have been cut-off from the area.

The class probabilities visualised in Figure 27b are as expected when taking into consideration the class scores. It is also visualised that the class probabilities are slowly converging towards the balance. This is due to the rising class score of class 1, which increases the relevance of the b parameter. This results are what we expect as we collect data farther from in-distribution data.

In Section 5.2 it became apparent that radial softmax might not learn the parameters correctly and thus would not be successful in limiting the area of confidence around the training dataset. In this case it is still able to learn and predict correctly.

We experimented with another use case for radial softmax, in which we fixed the parameters to values which suit our training dataset in Section 5.2.2. This experiment demonstrated that radial softmax with the correct parameters radial softmax can limit the areas of confidence for this dataset as well. This leads us to the conclusion that if the model does not learn the parameters correctly, a grid search on a possible space of parameters could lead to a successful use of radial softmax.

6 Future work

In this work we radial softmax had limited improvements over softmax for the real-life datasets. It also did not succeed in eight moons with two classes without fixing the learnable parameters to specific values. It therefore leads to further study on if this was the same case for the real-life datasets and if parameter tuning could lead to further improvements. This would however lead to increased training times, at least for when the tuning parameters are first discovered.

Another avenue of research is in increasing the probability that radial softmax learns the best b and c parameters for the data it is trained on. In the ablation study in Section 5 we saw that for the eight moons two classes dataset it failed in learning the correct parameters. Explaining why this happened is also an area of possible research.

7 Related Work

Baselines and metrics for detecting OOD examples by neural networks have been proposed for machine vision and different natural language processing subsets [3]. The work also discussed why softmax probabilities on their own are not suitable to be called model confidence due to there being distinct differences of probabilities between in-distribution and OOD examples without the model losing in true-positive vs. false-positive rates.

It has been shown that temperature scaling and small perturbations in input data leads to better performance in detecting OOD samples [31]. Their proposed method is called ODIN (Out-of-DIstribution detector for Neural networks) and can be used without retraining the model. Their method significantly improved performance on baseline results.

A bounded activation function has been developed to reduce open space risk [32]. The Tent activation function improves adversarial robustness by reducing open space. It has shown improvements on MNIST with an average accuracy of 91.8% on examples perturbed by adversarial white-box attacks, compared to 78.8% achieved with adversarial training with PGD.

It has been shown that ReLu networks widely produce high confidence predictions far away from the training data [4]. They propose a new optimisation technique which enforces low confidence predictions far from the training data while keeping high confidence predictions for the original prediction task. The proposed technique introduces efficiently generated adversarial images that structurally do not resemble in-distribution images.

Experiments have shown that it is possible to design reliable adversarial training methods which create adversarially robust models [13]. They have shown improvements on MNIST and that models with a higher capacity have increased robustness against one-step perturbations.

Nalisnick, E. et al [5] found that density learned by flow-based models are unable to distinguish datasets in pairings such as Fashion MNIST vs MNIST, ImageNet vs SVHN etc. They have shown that deep generative models are unable to identify if a datapoint is in the training set or of a similar input.

A deep verifier network (DVN) has been proposed to detect the unreliable inputs or predictions of deep generative models [33]. The verifier model is trained separately than the predictor model and is based on conditional variational auto-encoders. They show improvements in robustness to adversarial attacks on CIFAR-10, CIFAR-100 and SVHN. They provide baselines for out-of-distribution (OOD) detection for CUB, LSUN and COCO datasets.

An approach that can be used with ReLu networks and provides provably low confidence predictions far from the training data has been proposed [34]. They propose a model called certified certain uncertainty which guarantees close to uniform low

probability predictions for data points far from the training data.

8 Conclusion

In this thesis we defined a new proposed activation function based on softmax, called radial softmax. Radial softmax is designed to solve softmax's OOD problem, where one class will dominate over the other classes. This happens because the dominating class has a much higher class score than the others, leading to a very high class probability. To counteract this radial softmax is able to learn the approximate maximum probability for each class and the approximate class distribution with the aim of predicting based on the learned class distribution for OOD data.

For analysis we present the effects that radial softmax has on the areas of confidence and class scores and probabilities in comparison to softmax by using two moons and eight moons datasets. We demonstrated that radial softmax has significant improvements over softmax for these datasets.

We further demonstrated the effects by using real-life data to both train and evaluate the performance based on four metrics: testing error, MMC, AUROC and FPR@95. The tested models were trained on MNIST, SVHN, CIFAR-10 and CIFAR-100. Radial softmax performed in a comparable level to softmax in learning based on the test error. Based on the other three metrics there were improvements for most of the comparisons, but the improvements were small.

An ablation study was performed to gauge the performance on an eight moons two classes dataset. For this we reported results the same way as was done for the two and eight moons datasets. In addition we also reported the learned b and c values for both of the classes. In this study radial softmax did not perform as well as it did for two moons or eight moons. One of the reasons for that is that the b and c parameters were not correctly learned by the model. We performed a second ablation study where we fixed the b and c parameters before training the model on the eight moons two classes dataset. In this case the model performed as it was designed.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[2] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, et al. State-of-the-art Speech Recognition With Sequence-to-Sequence Models. *arXiv:1712.01769 [cs, eess, stat]*, February 2018.

[3] Dan Hendrycks and Kevin Gimpel. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. *arXiv:1610.02136 [cs]*, October 2018.

[4] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. *arXiv:1812.05720 [cs, stat]*, May 2019.

[5] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do Deep Generative Models Know What They Don't Know? *arXiv:1810.09136 [cs, stat]*, February 2019.

[6] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv:1811.03378 [cs]*, November 2018.

[7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety. *arXiv:1606.06565 [cs]*, July 2016.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[9] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv:1603.04467 [cs]*, March 2016.

[10] Giang Nguyen, Stefan Dlugolinsky, Martin Bobák, Viet Tran, Álvaro López García, et al. Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: A survey. *Artificial Intelligence Review*, 52(1):77–124, June 2019.

[11] François Chollet. Keras. `https://keras.io`, 2015. (14/04/2020).

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

[13] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv:1706.06083 [cs, stat]*, September 2019.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015.

[15] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, et al. Scikit-learn: Machine learning in python. *The Journal of machine Learning research*, 12:2825–2830, 2011.

[16] Patrick J. Grother. NIST Special Database 19. `https://www.nist.gov/srd/nist-special-database-19`, April 2019. (25/03/2020).

[17] Alex Krizhevsky. The CIFAR-10 dataset. `https://www.cs.toronto.edu/~kriz/cifar.html`. (25/03/2020).

[18] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747 [cs, stat]*, September 2017.

[19] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: An extension of MNIST to handwritten letters. *arXiv:1702.05373 [cs]*, March 2017.

[20] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv:1506.03365 [cs]*, June 2016.

[21] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, pages 1485–1488, New York, NY, USA, 2010. Association for Computing Machinery.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[23] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[24] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2285–2294, 2016.

[25] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. Learning multi-attention convolutional neural network for fine-grained image recognition. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2017.

[26] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity Mappings in Deep Residual Networks. *arXiv:1603.05027 [cs]*, July 2016.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017.

[29] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[30] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep Anomaly Detection with Outlier Exposure. *arXiv:1812.04606 [cs, stat]*, January 2019.

[31] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. *arXiv:1706.02690 [cs, stat]*, February 2018.

[32] Andras Rozsa and Terrance E. Boult. Improved Adversarial Robustness by Reducing Open Space Risk via Tent Activations. *arXiv:1908.02435 [cs]*, August 2019.

[33] Tong Che, Xiaofeng Liu, Site Li, Yubin Ge, Ruixiang Zhang, et al. Deep Verifier Networks: Verification of Deep Discriminative Models with Deep Generative Models. *arXiv:1911.07421 [cs]*, February 2020.

[34] Alexander Meinke and Matthias Hein. Towards neural networks that provably know when they don't know. *arXiv:1909.12180 [cs, stat]*, February 2020.

Appendix

I. Proofs

Let $\pi_1, \ldots, \pi_K \in (0, \infty)$ denote the class distribution and $\sum_{i=1}^{K} \pi_i = 1$.

Lemma 1. *Let $c > 0$ and $f(x) = \frac{x}{x+c}$. Then f(x) is strictly increasing in the range $x \in (0, \infty)$. That means $f(x') > f(x)$ for every $x' > x > 0$.*

Proof. Let us assume by contradiction that $f(x') < f(x)$, where $x' > x > 0$.

$$\begin{aligned} \frac{x'}{x'+c} &< \frac{x}{x+c} \qquad | * (x'+c)(x+c) \\ x'(x+c) &< x(x'+c) \\ x'c &< xc \\ x' &< x \end{aligned}$$

This is a contradiction. Thus $f(x') > f(x)$, where $x' > x > 0$. □

Theorem 2. *Let $t(\mathbf{z})$ and $s(\mathbf{z})$ be vectors with elements $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$, where $b_1, \ldots, b_k, c_1, \ldots, c_k \in \mathbb{R}$. Then $0 \leq s(\mathbf{z})_i \leq 1$, for $i = 1, \ldots, K$ and $\sum_{i=1}^{K} s(\mathbf{z})_i = 1$.*

Proof. First let us note that $t(\mathbf{z})_i > 0$ for any $i = 1, \ldots, K$. Therefore $0 < s(\mathbf{z})_i$. Since $t(\mathbf{z})_i < \sum_{j=1}^{K} t(\mathbf{z})_j$ we get that $s(\mathbf{z})_i \leq 1$.

We will next prove $\sum_{i=1}^{K} s(\mathbf{z})_i = 1$.

$$\sum_{i=1}^{K} \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{j=1}^{K} (e^{c_j - |z_j|} + e^{b_j})} = \frac{\sum_{i=1}^{K} (e^{c_i - |z_i|} + e^{b_i})}{\sum_{j=1}^{K} (e^{c_j - |z_j|} + e^{b_j})} = 1$$

□

Theorem 3. *Let $t(\mathbf{z})$ and $s(\mathbf{z})$ be vectors with elements $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$, where $b_1, \ldots, b_k, c_1, \ldots, c_k \in \mathbb{R}$. Then for each i and for any real*

values $z_1, \ldots, z_{i-1}, z_{i+1}, \ldots z_K$*, the function* $s(\mathbf{z})_i$ *as a function of* z_i *is symmetrical around* $z_i = 0$*, strictly decreasing as* $z_i \to \infty$ *and maximised at* $z_i = 0$*.*

Proof. For any $\mathbf{z}$ we define $\mathbf{z}'$ such that each element of $\mathbf{z}'$ is the same as the corresponding element of $\mathbf{z}$, except for $\mathbf{z}'_i = -\mathbf{z}_i$. Symmetry of $s(\mathbf{z})_i$ follows from the fact that $|-z_i| = |z_i|$. Thus $s(\mathbf{z})_i = s(\mathbf{z}')_i$.

We show that $s(\mathbf{z})_i$ is strictly increasing in the range $(-\infty, 0)$ and decreasing in the range $(0, \infty)$. Due to the symmetry property we must prove only one. For any $\mathbf{z}$ we define $\mathbf{z}''$ such that each element of $\mathbf{z}''$ is the same as the corresponding element of $\mathbf{z}$, except $z''_i > z_i > 0$. Using Lemma 1, where $x = t(\mathbf{z})_i$ and $x' = t(\mathbf{z}'')_i$ and $c = \sum_{\substack{j=1 \\ j \neq i}}^{K} (e^{c_j - |z_j|} + e^{b_j})$, we get $t(\mathbf{z})_i > t(\mathbf{z}'')_i > 0$. From this follows $s(\mathbf{z})_i > s(\mathbf{z}''_i)$ and that $s(\mathbf{z}_i)$ is max at $z_i = 0$.

□

Theorem 4. *Let* $t(\mathbf{z})$ *and* $s(\mathbf{z})$ *be vectors with elements* $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ *and* $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$*, where* $b_i = \ln \pi_i$ *and* $c_1, \ldots, c_k \in \mathbb{R}$*. Then for* $\epsilon > 0$ *and for each* $z_1, \ldots, z_K$*, such that* $|z_i| > M$*, it follows that* $|s(\mathbf{z})_i - \pi_i| < \epsilon$*.*

Proof. First let us note that for each i:

$$\lim_{|z_i| \to \infty} e^{c_i - |z_i|} + e^{b_i} = e^{b_i} = \pi_i$$

Thus there exists an M, such that if for each $z_1, \ldots, z_K$ and $|z_i| > M$ then $|t(\mathbf{z})_i - \pi_i| < \frac{\epsilon}{K}$. We can rewrite it as $-\frac{\epsilon}{K} + \pi_i < t(\mathbf{z})_i < \frac{\epsilon}{K} + \pi_i$. We will show that from $t(\mathbf{z})_i < \frac{\epsilon}{K} + \pi_i$, it follows that $|s(\mathbf{z})_i - \pi_i| < \epsilon$. First we will look at $s(\mathbf{z})_i < \epsilon + \pi_i$.

Let us separately look at the numerator and denominator. The numerator becomes $e^{c_i - |z_i|} + e^{b_i} < \frac{\epsilon}{\pi_i} + \pi_i$. The denominator is $\sum_{j=1}^{K} (e^{c_j - z_j} + e^{b_j}) = \sum_{j=1}^{K} (e^{c_j - z_j}) + 1 > 1$. As such we need to consider the numerators, thus $s(\mathbf{z})_i < \epsilon + \pi_i$.

We now prove $\pi_i - \epsilon < s(z)_i$. Note since that $t(\mathbf{z})_i > \pi_i$:

$$s(z)_i = \frac{t(z_i)}{\sum_{j=1}^{K} t(z_j)} = \frac{(t(z_i) - \pi_i) + \pi_i}{\sum_{j=1}^{K} (t(z_j) - \pi_j) + \pi_j} > \frac{\pi_i}{\sum_{j=1}^{K} \frac{\epsilon}{k} + \pi_j} = \frac{\pi_i}{\epsilon + 1}$$

This can be proved to be greater than $\pi_i - \epsilon$ as follow:

$$\frac{\pi_i}{\epsilon+1} > \pi_i - \epsilon$$
$$\pi_i > \pi_i\epsilon + \pi_i - \epsilon^2 - \epsilon$$
$$0 > \pi_i - \epsilon - 1$$

Thus $|s(z)_i - \pi_i| < \epsilon$. □

Theorem 5. *For each $M_1, M_2, \ldots, M_k \in (0,1)$, where $M_i > \pi_i$, there exists $b_1, \ldots, b_K, c_1, \ldots, c_K \in \mathbb{R}$, such that for each i, $\sup_{\mathbf{z}} s(\mathbf{z})_i = M_i$. Where $t(\mathbf{z})_i = e^{c_i - |z_i|} + e^{b_i}$ and $s(\mathbf{z})_i = \frac{t(\mathbf{z})_i}{\sum_{j=1}^{K} t(\mathbf{z})_j}$.*

Proof. Let us define $b_i = \ln \pi_i$, $c_i = \ln \frac{M_i - \pi_i}{1 - M_i}$ and a vector $\mathbf{z}'_A$, where each element is equal to A except for the ith element, which is equal to 0. We prove that $s(\mathbf{z}'_A)_i$ is strictly increasing over A.

$$s(\mathbf{z}'_A)_i = \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{j=1}^{K} (e^{c_j - |z_j|} + e^{b_j})} = \frac{e^{c_i} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} (e^{c_j - |A|} + e^{b_j}) + e^{c_i} + e^{b_i}} \tag{4}$$

In Equation 4 as A increases, the numerator stays the same and the denominator decreases, thus $s(\mathbf{z}'_A)_i$ is strictly increasing over A.

We next show that $\lim_{A \to \infty} s(\mathbf{z}'_A)_i = M_i$.

$$\lim_{A \to \infty} s(\mathbf{z}'_A) = \lim_{A \to \infty} \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} (e^{c_j - |z_j|} + e^{b_j}) + e^{c_i - |z_i|} + e^{b_i}} = \frac{e^{c_i} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} (e^{c_j - |z_j|} + e^{b_j}) + e^{c_i} + e^{b_i}}$$
$$= \frac{e^{c_i} + e^{b_i}}{1 - e^{b_i} + e^{c_i} + e^{b_i}} = \frac{\frac{M_i - \pi_i}{1 - M_i} + \pi_i}{1 + \frac{M_i - \pi_i}{1 - M_i}} = \frac{\frac{M_i - \pi_i + \pi_i - \pi_i M_i}{1 - M_i}}{\frac{1 - M_i + M_i - \pi_i}{1 - M_i}} = \frac{\frac{M_i(1 - \pi_i)}{1 - M_i}}{\frac{1 - \pi_i}{1 - M_i}}$$
$$= \frac{M_i(1 - \pi_i)(1 - M_i)}{(1 - M_i)(1 - \pi_i)} = M_i$$

It remains to show that for any $\mathbf{z}$ we have that $s(\mathbf{z})_i < s(\mathbf{z}'_A)_i < M_i$, where $A = max(|z_1|, |z_2|, \ldots, |z_k|)$.

$$s(\mathbf{z}'_A)_i = \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{j=1}^{K} \left(e^{c_j - |z_j|} + e^{b_j}\right)} = \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} \left(e^{c_j - |z_j|} + e^{b_j}\right) + e^{c_i - |z_i|} + e^{b_i}}$$
$$\leq \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} \left(e^{c_j - |A|} + e^{b_j}\right) + e^{c_i - |z_i|} + e^{b_i}}$$

Using Lemma 1, where $x = e^{c_i - |z_i|} + e^{b_i}$ and $c = \sum_{\substack{j=1 \\ j \neq i}}^{K} \left(e^{c_j - |A|} + e^{b_j}\right)$ we can continue the sequence of inequalities:

$$s(\mathbf{z}'_A)_i \leq \frac{e^{c_i - |z_i|} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} \left(e^{c_j - |A|} + e^{b_j}\right) + e^{c_i - |z_i|} + e^{b_i}} \leq \frac{e^{c_i} + e^{b_i}}{\sum_{\substack{j=1 \\ j \neq i}}^{K} \left(e^{c_j - |A|} + e^{b_j}\right) + e^{c_i} + e^{b_i}} = s(\mathbf{z}'_A)_i$$

We have shown that $\lim_{A \to \infty} s(\mathbf{z}'_A)_i = M_i$. Thus for any $M_1, M_2, \ldots, M_k$ there exists $b_1, \ldots, b_K, c_1, \ldots, c_K \in \mathbb{R}$ such that for any i it follows $\sup_{\mathbf{z}} s(\mathbf{z})_i = M_i$.

□

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Rain Vagel**,
(author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Radial Softmax: A Novel Activation Function for Neural Networks to Reduce Overconfidence in Out-Of-Distribution Data,
   (title of thesis)

   supervised by Ardi Tampuu, Meelis Kull and Raul Vicente.
   (supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Rain Vagel
15/05/2020