Using Reed-Muller Codes for Classification with Rejection and Recovery

Daniel Fentham[1[0009−0000−2907−356X]], David Parker[2[0000−0003−4137−8862]], and Mark Ryan[1[0000−0002−1632−497X]]

[1] School of Computer Science, University of Birmingham, Birmingham, United Kingdom `dxf209@student.bham.ac.uk`, `m.d.ryan@cs.bham.ac.uk`

[2] Department of Computer Science, University of Oxford, Oxford, United Kingdom `david.parker@cs.ox.ac.uk`

Abstract. When deploying classifiers in the real world, users expect them to respond to inputs appropriately. However, traditional classifiers are not equipped to handle inputs which lie far from the distribution they were trained on. Malicious actors can exploit this defect by making adversarial perturbations designed to cause the classifier to give an incorrect output. Classification-with-rejection methods attempt to solve this problem by allowing networks to refuse to classify an input in which they have low confidence. This works well for strongly adversarial examples, but also leads to the rejection of weakly perturbed images, which intuitively could be correctly classified. To address these issues, we propose Reed-Muller Aggregation Networks (RMAggNet), a classifier inspired by Reed-Muller error-correction codes which can correct and reject inputs. This paper shows that RMAggNet can minimise incorrectness while maintaining good correctness over multiple adversarial attacks at different perturbation budgets by leveraging the ability to correct errors in the classification process. This provides an alternative classification-with-rejection method which can reduce the amount of additional processing in situations where a small number of incorrect classifications are permissible.

Keywords: Deep Neural Networks · Adversarial Examples · Classification-with-rejection · Error-correction codes · ML Security

1 Introduction

Deep Neural Networks (DNNs) have shown incredible performance in numerous classification tasks, including image classification [1], medical diagnosis [2] and malware detection [3]. However, a fundamental shortcoming is that they pass judgement beyond their expertise. When presented with data outside of the distribution they were trained on, DNNs will attempt to classify that data by selecting from one of the finite labels available, often reporting high confidence in the classifications they have made. The most egregious examples of this occur when a DNN is presented with an input which is far outside of the domain it has been trained on (for example, presenting an image of a cat to a text classification model), which it will confidently assign a class to. This behaviour is also present in adversarial examples, which were introduced in the seminal paper by Szegedy et al. in 2018 [4], where an (almost) invisible perturbation pushes an input far from the training distribution, leading to a confident misclassification [5]. Since then, extensive research has been conducted exploring new, more sophisticated, attacks on networks of different architectures [6,7,8] and defences that attempt to mitigate their effectiveness [9,10,11]. This results in hesitation when applying DNN models to safety- and security-critical applications where there is a high cost of misclassification.

Classification-with-rejection (CWR) methods [11,12,13,14] attempt to address this limitation by refusing to assign a label to an input when the confidence in the classification is low. In this paper, we present an approach to CWR that parallels ideas from Error-Correcting Output Codes (ECOCs) [11,14], where an ensemble of networks perform classification by generating binary strings, extending them with a reject option. ECOC methods have received little attention as a defence mechanism against adversarial attacks, even though the independence of the individual networks offers a natural defence. A notable property of adversarial attacks is that they are highly transferable between models, meaning an adversarial attack crafted for one network will likely deceive another with high probability, provided the networks perform a similar task [15]. Since ECOC methods promote diversity in the constituent network's tasks, an adversarial attack crafted to change the output reported by one network is less likely to fool another in a way which would result in further misclassification. Moreover, due to the aggregated nature of the resulting classification, an adversary would need to create a perturbation which can fool multiple networks simultaneously, necessitating precise bit-flipping strategies which lead to a valid target class.

In this paper, we introduce Reed-Muller Aggregation Networks (RMAggNet), which apply error correcting codes to the correction and rejection of classifications, ultimately producing a new kind of ECOC classifier. Similar to existing ECOCs, these consist of multiple DNNs, each performing a simple classification task which determines if an input belongs to a defined subset of the classes, resulting in a binary answer. The results of these networks are aggregated together into a binary string which we compare to class binary strings which represent each of the classes from the dataset. If the resulting binary string is the same as a class binary string, we return the associated label as a result, otherwise we attempt to correct the result (if we have a small enough Hamming distance), or reject the result and refuse to classify the input. Thus, unlike existing CWR methods our approach has the ability to both correct and reject inputs.

We evaluate the effectiveness of RMAggNet by comparing it to two other CWR approaches: an ensemble of networks (with a voting-based rejection process) and Confidence Calibrated Adversarial Training (CCAT) [16]. After performing tests using the EMNIST and CIFAR-10 datasets with open-box PGD L_∞ and PGD L_2 adversarial attacks, we conclude that RMAggNet can greatly reduce the amount of rejected inputs in certain circumstances, making it a viable alternative to methods such as CCAT if some incorrectness is acceptable. We expand on this finding using the MNIST dataset, along with closed-box adversarial attacks, in an extended version of this paper [17].

In summary, this paper makes the following contributions:

– We introduce RMAggNet, a novel ECOC classification method which leverages the power of Reed-Muller codes to create a classifier which can both correct and reject inputs (Section 3)[3].
– We show the effectiveness of RMAggNet on the EMNIST and CIFAR-10 datasets with open-box gradient-based adversarial attacks (Sections 4 & 5).
– We discuss the application of RMAggNet to classification tasks, providing guidance on when it may be a strong alternative to other CWR methods (Section 6).

2 Related work

2.1 Error correcting output codes

Verma and Swami defined an error-correcting output code (ECOC) classification method which uses an ensemble of models, each trained to perform a subset of the classification [11]. Their model uses Hadamard matrices to construct binary codes, which are assigned to classes from the dataset. Multiple DNNs are then defined to generate a set amount of bits from each code for each class, essentially following a set membership classification approach. When an input is passed to the multiple networks, a vector of real numbers is generated, and the similarity between this vector and the class vectors is calculated, with the most similar vector being returned as the final classification.

The authors argue that this classification method has greater resilience to adversarial attacks than traditional ensemble methods due to the independence of the models, encouraged by the diverse classification tasks. This reduces the chance of multiple coordinated bit-flips occurring due to a single perturbation as a result of the transferability of adversarial attacks. Verma and Swami focus their attention on multi-bit outputs, where four networks produce a combined total of 16, 32, or 64 bits, encoding the input into 4, 8 or 16 bits, respectively. This approach to ECOC classification leads to similar networks being trained, where, in many cases, the entire set of classes is being used by all networks. This results in each network learning similar features, reducing independence and lowering resilience to transfer attacks.

Song et al., proposed a method which extends the work by Verma and Swami, introducing Error Correcting Neural Networks (ECNN) [14]. This paper improves ECOCs by increasing the number of networks to one per output bit and optimising the codeword matrix using simulated annealing [18] which encourages each classifier to learn unique features, enhancing robustness against direct and transfer adversarial attacks. However, in practice, the ECNN implementation trains a single network with each classifier having a unique top layer. This reduces

[3] Code available at: `https://github.com/dfenth/RMAggNet`

the independence between networks since each of them share the same low level features which can be used by adversaries.

These approaches are similar to the method proposed in this paper; however, there are a few key differences. While Verma and Swami [11] and Song et al. [14] discuss the use of error correction, it is not actively utilised in the classification process. In addition, error correction provides a natural implementation of CWR where outputs which deviate significantly from existing classes can trigger a *reject* option where the classifier refuses to return a result. This paper aims to address these gaps by exploring the application of error correction and classification-with-rejection approaches to ECOC methods. We hope to provide insights into the effectiveness, practicality and benefits of these strategies.

2.2 Confidence Calibrated Adversarial Training

Many CWR methods have been proposed over the years [12,13,16]. We focus on the Confidence Calibrated Adversarial Training (CCAT) CWR method which was introduced by Stutz et al. in 2020 [16]. CCAT attempts to produce a model which is robust to unseen threat models which use different L_p norms or larger perturbations when generating adversarial examples. CCAT achieves good rejection performance through adversarial training where the model is trained to predict the classes of clean data with high confidence and produce a uniform distribution for adversarial examples within an ϵ-ball of the true image.

3 Reed-Muller Aggregation Networks (RMAggNet)

3.1 Reed-Muller codes

We begin with some brief background on Reed-Muller codes, which are multi-error detecting and correcting codes [19,20]. This extends earlier work on Hamming codes [21] and generalise many other error correction methods.

Reed-Muller is often represented with the notation $[2^m, k, 2^{m-r}]_q$. We set q, the number of elements in the finite field, to 2, meaning any codes we create will be binary. In the low-degree polynomial interpretation, m denotes the number of variables and r denotes the highest degree of the polynomial both of which influence the properties of the Reed-Muller code. The first element of the tuple (2^m) represents the length of the codewords we will use. The second element (k) represents the length of the message we can encode and is calculated as

$$k = \sum_{i=0}^{r} \binom{m}{i} \tag{1}$$

The final element (2^{m-r}) is the minimum Hamming distance between any two codes we generate, and influences the amount of error correction that can be applied. For simplicity, we set $n = 2^m$ and $d = 2^{m-r}$ condensing the notation to $[n, k, d]_2$.

A key advantage of Reed-Muller codes is that they allow us to unambiguously correct a number of bits equal to the Hamming bound t:

$$t = \left\lfloor \frac{(d-1)}{2} \right\rfloor \tag{2}$$

due to the guaranteed Hamming distance between any two codewords. This can be thought of as an open Hamming sphere around each codeword with a radius of 2^{m-r-1} which does not intersect any other sphere.

To build Reed-Muller codes with pre-determined Hamming distances, we start by selecting values for m and r which fit our use case, i.e., we can generate codewords of appropriate length with a desired amount of correction. Once we have chosen m and r, we can calculate k (see equation 1) and we can define the low-degree polynomial, which will have k coefficients, m variables and a maximum degree of r. This allows us to generate the codewords with a minimal Hamming distance of d between any two of the codes.

We specify the coefficients of the polynomial in k different ways where a single coefficient is set to one, and all others are zero. This gives us k polynomials with fixed coefficients and m free variables. We can then define the basis vectors of the space by instantiating every possible combination of variables for each of the fixed coefficient polynomials. This creates k codewords of length 2^m all of which have a guaranteed Hamming distance of at least d. We can have up to 2^k valid codewords which satisfy the Hamming distance guarantee. These additional codewords can be generated by performing an XOR operation on all possible combinations of the basis vectors generating a closed set. We refer to these binary vectors as *codewords*.

3.2 Reed-Muller Aggregation Networks

We can now define a Reed-Muller Aggregation Network (RMAggNet) which uses multiple networks, trained on separate tasks, to create a binary vector which we can classify, correct, or reject. To create an RMAggNet we start by defining Reed-Muller codes which act as class codewords for the dataset classes we intend to recognise. To define appropriate Reed-Muller codes, we have to consider a number of factors related to the problem we are solving.

The first is the number of classes in the dataset ($|C|$). We must make sure that the message length k is adequate for the number of classes, such that, $|C| \leq 2^k$. From the definition of k (equation 1) we can see that it depends on both m and r, therefore these values are influenced by $|C|$ and must be considered early on in the design process. The number of classes in the dataset is the primary point to consider when deciding on values for m and r, because if we do not satisfy $|C| \leq 2^k$ then we will not have an effective classifier.

The second factor we must consider, is whether we have appropriate error correction for the problem. The maximum number of errors we can correct is represented by the Hamming bound t (equation 2) which relies on d which is 2^{m-r}, so we also need to take this into account when deciding on m and r.

The third factor is that we must have a low probability of assigning a valid codeword to a random noise image. A fundamental flaw with traditional DNNs is that they will assign a class to any input, even if the input is far from the distribution they have been trained on. The probability of assigning a random noise image a valid class is $|C|/2^n$ where we have no error correction, however, with error correction, the probability increases to $(|C| \cdot \sum_{i=0}^{t} \binom{n}{i})/2^n$. This means that it is advantageous to only use the amount of error correction that is necessary for the problem at hand, even if that means we are not correcting the maximum number of bits theoretically possible.

Once we have a set of codewords which fit the problem specification, the length of the codewords (n) determines the number of networks included in the aggregation. We can assign each network a unique index value corresponding to an index within the class codeword binary strings. The values at the index positions define a set partition which determines which classes a network is trained to return a 1 for and which it returns a 0 for (i.e., the network returns a 1 for all classes with a 1 at the index and 0 otherwise). We can also view the class codewords as a matrix, with each network assigned to a column with 1s and 0s indicating the partition between sets. By randomly shuffling the class codewords, each network is trained to recognise a set of approximately half of the classes.

With the classification task for each network defined, we can move on to training. The training process requires us to adjust the true labels of each input with each network having a unique set of true labels for the training data, where the true labels correspond to the set partition label. Once the dataset has been adjusted each network is trained independently.

During inference we pass the same input to the n networks which produces n real values $\mathbf{v} \in \mathbb{R}^n$. We select a threshold value τ which acts as a bias, with a large τ leading to codewords consisting of more 0 bits. We compare each of the n real values of $\mathbf{v}$ to τ with the following rule:

$$v_i = \begin{cases} 1 & \text{if } v_i \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

This produces a binary string which we can compare to the class codewords. If any of the class codewords match the predicted binary string exactly, we can return the label associated with it as the result; however, if none match, we calculate the Hamming distance between the prediction and the class codewords. If we find a Hamming distance less than or equal to t, then we can unambiguously correct to, and return, that class codeword due to the properties of Reed-Muller codes. Otherwise, we refuse to classify the input and return a rejection.

4 Evaluation methodology

4.1 Threat model

We begin by establishing the threat model under which we expect the RMAggNet defence to operate effectively as per the recommendations made by Carlini et al. [22]. Our assumptions consider an adversary who knows the purpose of the model (i.e., the classes the model can output) and is capable of providing the model with inputs. The actions of the adversary are constrained by a limited perturbation cost, where the L_p-norm between the original (x) and perturbed ($\tilde{x}$) image must be below some threshold ϵ, i.e., ($|x - \tilde{x}|_p \leq \epsilon$), where p is either 2 or ∞ depending on the attack used. The norm used for an attack changes the focus of the adversarial perturbation. The L_2 attacks encourage small changes across all input dimensions, which distributes the perturbation across multiple features, whereas the L_∞ attack encourages perturbations which focus on a single feature, maximising this as much as the budget (ϵ) allows. The ultimate aim of the adversary is to generate perturbed images which are not noticeable to a time constrained human.

The level of access to the model granted to the adversary depends on the specific adversarial attack being employed. In the case of open-box attacks, the adversary has full access to the target model and can generate adversarial perturbations tailored to deceive that particular network. This represents the worst-case scenario. On the other hand, closed-box attacks represent a more realistic setting where the adversary does not have access to the model parameters. In this case it is necessary to train a surrogate model which performs the same classification task as the target model. Adversarial examples are then generated for this surrogate model, which leverage transferability to create adversarial examples for the target model. In this paper we focus on the more challenging open-box attacks. Experiments using closed-box attacks can be found in the extended version of this paper [17].

We employ two attacks to generate adversarial examples, focusing on the gradient-based Projected Gradient Descent (PGD) attack [23], using both the L_2- and L_∞-norm. In the extended paper we also include the gradient-free Boundary attack in the L_2-norm [24], and transfer attacks to demonstrate the adversarial robustness of these approaches in a closed-box setting. The use of both gradient and gradient-free attacks allows us to more thoroughly evaluate the robustness of the models. While gradient based attacks tend to produce stronger adversarial examples, they can fail to produce effective perturbations if the target model performs any kind of gradient masking. To ensure that we have a fair and reliable evaluation of the robustness we include gradient-free attacks to eliminate the possibility that any results are solely due to masking the model gradients.

4.2 Comparison methods

To evaluate the classification and rejection ability of RMAggNet we have implemented two other comparison methods.

The first is a traditional ensemble method which consists of n networks (where n is the same number of networks used for RMAggNet) each of which are trained to perform the full classification task, as opposed to RMAggNet where each network is trained to perform set membership over two partitions. To aggregate the ensemble method results from the multiple networks, we have set up a simple voting system with an associated threshold (σ). When an input is passed to the ensemble, each network classifies the data producing a predicted class. If we exceed the threshold with the percentage of networks that agree on a single class, that class is returned as the most likely answer, otherwise, the input is rejected and no class is returned.

The second is the CCAT method (see Section 2.2) which uses adversarial training as per the original paper [16] using the original code which is slightly modified [4]. The adversarial training process allows CCAT to reject adversarial inputs within an ϵ-ball by learning to return a uniform distribution over all of the classes. This is then extrapolated beyond the ϵ-ball to larger perturbations. A threshold (τ) is specified which represents the confidence bound that must be exceeded so that the result is not rejected. Unlike the original paper where an optimal τ is calculated based on performance on the clean dataset, we vary τ to determine the effect on the rejection ability.

4.3 Datasets

We use multiple datasets to evaluate the effectiveness of RMAggNet on a variety of classification tasks. We focus on the EMNIST (balanced) [25] and CIFAR-10 datasets. The EMNIST dataset provides us with a simple classification task which consists of 131,600 grey-scale images of size 28×28, with 47 balanced classes including handwritten digits, upper- and lower-case letters (with some lower case classes excluded). Since we have 47 classes, the number of networks in RMAggNet is expanded to 32, with the same amount used for the Ensemble method. CIFAR-10 represents a more challenging classification task, increasing the image complexity with full colour images of size 32×32 over 10 possible classes which uses 16 networks for RMAggNet and Ensemble.

4.4 Generation of adversarial examples

To generate adversarial images from the selected datasets we use the FoolBox library [26,27]. We generate adversarial images using PGD L_2 and PGD L_∞ attacks. Due to the complex nature of some of the networks, adjustments needed to be made to generate adversarial examples.

RMAggNet: Due to the non-differentiable nature of RMAggNet from thresholding, direct attacks are difficult to generate. Following approaches such as BPDA [28] we implement a hybrid RMAggNet which replaces the final mapping from a binary string to class (or reject) with a Neural Network. This allows us to backpropagate through the entire model to produce effective adversarial examples.

Ensemble: We implement an ensemble via logits method [29] where the result of each network is weighted. Due to the voting system for the rejection we set equal weights over all networks. This approach allows us to have an ensemble method which mimics the voting output, except it is differentiable, therefore we can generate adversarial examples using the multiple networks directly.

[4] `https://github.com/davidstutz/confidence-calibrated-adversarial-training`

CCAT: Since CCAT is a standard Network which has undergone specific adversarial training, the generation of adversarial attacks is simple. Many attacks in the FoolBox library are able to generate adversarial examples without any modification of the network.

5 Results

5.1 EMNIST Dataset

Results for the EMNIST dataset use RMAggNet with $m = 5$, $r = 1$ which gives us 32 networks with 7 bits of error correction (EC). We also use 32 networks for the Ensemble method for parity. All methods use ResNet-18 models [30]. Table 1 shows the results on the clean EMNIST dataset where we expect to maximise correctness. All three models perform similarly, with Ensemble achieving the highest correctness, closely followed by RMAggNet and CCAT. However, all models come close to state-of-the-art performance (91.06%) [31], with minimal negative impacts from the adversarial defence.

Table 1: Results for the clean EMNIST dataset showing the percentage of classifications that are correct, rejected and incorrect. Bold text indicates the metric of interest. Higher correctness is better.

CCAT			
τ	**Correct**	**Rejected**	**Incorrect**
0	**88.68**	0.00	11.32
0.10	**88.60**	0.15	11.24
0.20	**87.54**	2.41	10.05
0.30	**85.46**	6.45	8.09
0.40	**83.16**	10.29	6.55
0.50	**80.70**	14.22	5.08
0.60	**77.95**	18.03	4.02
0.70	**74.23**	22.71	3.06
0.80	**69.48**	28.46	2.06
0.90	**60.74**	38.05	1.20
1.0	**0.00**	100.00	0.00

Ensemble			
σ	**Correct**	**Rejected**	**Incorrect**
0	**89.78**	0.00	10.22
0.10	**89.78**	0.00	10.22
0.20	**89.78**	0.01	10.21
0.30	**89.76**	0.05	10.19
0.40	**89.70**	0.28	10.03
0.50	**89.20**	1.41	9.39
0.60	**87.76**	4.45	7.79
0.70	**85.94**	7.68	6.38
0.80	**83.89**	11.05	5.06
0.90	**80.60**	15.60	3.80
1.0	**68.34**	30.15	1.51

RMAggNet $[32, 6, 16]_2$			
EC	**Correct**	**Rejected**	**Incorrect**
7	**89.16**	2.14	8.70
6	**87.93**	4.59	7.48
5	**86.54**	6.98	6.48
4	**84.99**	9.49	5.52
3	**83.29**	12.03	4.68
2	**80.85**	15.15	4.00
1	**77.19**	19.59	3.23
0	**70.02**	27.72	2.26

The performance on adversarial datasets generated with open-box attacks is shown in Tables 2 and 3. For both of these experiments we aim to minimise incorrectness, either by correctly classifying or rejecting the data. Correctly classifying the data is preferred since it reduces the reliance on downstream rejection handling.

Table 2 shows the results of the PGD L_∞ attack at varying perturbation budgets (ϵ). The CCAT results demonstrate strong performance with 0% incorrectness for $\tau > 0$ for all ϵ. At $\tau = 0$ we effectively disable the confidence threshold of CCAT and see 100% incorrectness as it is trained to return a uniform distribution for adversarial inputs. It is worth noting that the 0% incorrectness of CCAT is achieved through the rejection of all of the inputs, even at lower ϵ where both Ensemble and RMAggNet show that correct classifications can be recovered. This points towards a disadvantage of the conservative nature of CCAT. In situations where we want the option to reject, but can tolerate some incorrectness, CCAT often becomes ineffective for classification. Comparing Ensemble and RMAggNet, RMAggNet can achieve significantly lower incorrectness over all ϵ, translating the incorrectness into correct or rejected classifications depending on the amount of EC. This leads to RMAggNet being able to achieve higher correctness than both Ensemble and CCAT.

The results of the PGD L_2 attacks are in Table 3. The results are similar to those in Table 2, with CCAT reducing incorrectness to 0% through rejection alone. RMAggNet outperforms Ensemble in both maximum correctness and minimum incorrectness for $\epsilon = 0.30$ and $\epsilon = 1.0$. However, Ensemble can achieve higher correctness for $\epsilon = 3.0$ at the cost of incorrectness, which remains significantly higher than RMAggNet's.

From these results, we can conclude that RMAggNet provides more flexibility where small amounts of incorrectness is tolerable. The error correction process allows us to make trade-offs between maximising correctness and minimising incorrectness, with RMAggNet outperforming Ensemble in both of these metrics. RMAggNet comes close to CCAT in minimising incorrectness with the added advantage that, for small ϵ, we can recover and correctly classify many of the inputs, reducing pressure on downstream rejection handling.

Table 2: Results of PGD L_∞ adversaries generated using open-box attacks on EMNIST images with percentages of correct, rejected and incorrect classifications. Lower incorrectness is better.

PGD(L_∞)												
CCAT												
τ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	0.00	0.00	**100.00**	0.10	0.00	0.00	**100.00**	0.30	0.00	0.00	**100.00**
0.30		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
0.70		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
0.90		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
Ensemble												
σ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	71.70	0.00	**28.30**	0.10	29.40	0.00	**70.60**	0.30	0.00	0.00	**100.00**
0.30		71.70	0.00	**28.30**		29.40	0.00	**70.60**		0.00	0.00	**100.00**
0.70		64.10	15.20	**20.70**		20.80	22.60	**56.60**		0.00	0.00	**100.00**
1.00		31.40	60.10	**8.50**		3.20	73.70	**23.10**		0.00	4.60	**95.40**
RMAggNet												
EC	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
7	0.05	80.70	3.60	**15.70**	0.10	62.90	8.70	**28.40**	0.30	0.50	39.50	**60.00**
6		78.00	8.50	**13.50**		59.30	15.60	**25.10**		0.40	54.00	**45.60**
5		75.00	13.70	**11.30**		54.10	23.40	**22.50**		0.30	68.30	**31.40**
4		71.90	17.90	**10.20**		48.20	31.50	**20.30**		0.10	77.80	**22.10**
3		68.90	22.20	**8.90**		41.40	41.50	**17.10**		0.10	87.90	**12.00**
2		63.20	29.10	**7.70**		29.30	56.70	**14.00**		0.00	94.40	**5.60**
1		54.30	39.60	**6.10**		16.80	72.40	**10.80**		0.00	98.50	**1.50**
0		28.50	67.10	**4.40**		2.30	92.00	**5.70**		0.00	100.00	**0.00**

Table 3: Results of PGD L_2 adversaries generated using open-box attacks on EMNIST images with percentages of correct, rejected and incorrect classifications. Lower incorrectness is better.

PGD(L_2)													
CCAT													
τ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.30	0.20	0.00	**99.80**	1.0	0.00	0.00	**100.00**	3.0	0.00	0.00	**100.00**
0.30		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
0.70		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
0.90		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
Ensemble												
σ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.30	84.80	0.00	**15.20**	1.0	62.40	0.00	**37.60**	3.0	19.60	0.00	**80.40**
0.30		84.80	0.00	**15.20**		62.40	0.00	**37.60**		19.60	0.00	**80.40**
0.70		79.50	9.10	**11.40**		57.10	13.00	**29.90**		19.50	0.20	**80.30**
1.00		60.60	35.90	**3.50**		47.40	39.80	**12.80**		18.50	9.80	**71.70**
RMAggNet												
EC	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
7	0.30	86.00	3.10	**10.90**	1.0	70.40	6.60	**23.00**	3.0	9.20	25.70	**65.10**
6		84.40	6.20	**9.40**		67.60	12.20	**20.20**		6.40	37.60	**56.00**
5		82.70	9.10	**8.20**		65.20	17.40	**17.40**		5.00	45.60	**49.40**
4		80.70	12.40	**6.90**		61.70	23.40	**14.90**		3.10	55.60	**41.30**
3		77.70	16.40	**5.90**		57.40	29.30	**13.30**		2.00	65.00	**33.00**
2		74.10	20.90	**5.00**		50.30	38.10	**11.60**		1.00	75.60	**23.40**
1		68.00	28.20	**3.80**		39.10	51.50	**9.40**		0.70	86.10	**13.20**
0		56.00	41.30	**2.70**		15.10	78.60	**6.30**		0.00	94.60	**5.40**

5.2 CIFAR-10 Dataset

Results for the CIFAR-10 dataset use RMAggNet with $m = 4$, $r = 1$ which gives us 16 networks with 3 bits of error correction. We use 16 networks in the Ensemble method for parity. All networks use an architecture outlined in the extended paper.

Table 4 shows the results on the clean CIFAR-10 dataset where we aim to maximise correctness. Ensemble reports the highest correctness, followed by RMAggNet, then CCAT. All models report correctness within 3% of one another, so, we can conclude that all are equally capable in terms of classification ability.

Table 5 shows the results for the L_∞ attacks on CIFAR-10, with Table 5a reporting the results of the open-box attack on the surrogate model. This indicates that the attack on the surrogate model is very effective, leading to low accuracy of the model at all $\epsilon > 0$. The L_∞ closed-box transfer attacks on the CIFAR-10 models can be seen in Table 5b, where we aim to reduce incorrectness. The results show a strong adversarial attack due to low correctness from Ensemble and RMAggNet, even for low values of ϵ. The CCAT model is able to reliably reject all adversarial inputs for all non-zero confidence thresholds. The Ensemble method classifies significantly more inputs correctly compared to RMAggNet for all values of ϵ that we tested, however, the percentage of correct classifications are still low, leading to an ineffective classifier. This result shows that CCAT is an effective method for avoiding adversarial results when strong attacks are used. Further results from closed-box transfer attacks can be found in the full paper [17].

Table 4: Results for the clean CIFAR-10 dataset showing the percentage of classifications that are correct, rejected and incorrect. Bold text indicates the metric of interest. Higher correctness is better.

CCAT			
τ	**Correct**	**Rejected**	**Incorrect**
0	**76.41**	0.00	23.59
0.10	**76.41**	0.00	23.59
0.20	**76.21**	0.68	23.11
0.30	**75.10**	3.76	21.14
0.40	**72.62**	9.38	18.00
0.50	**68.59**	17.41	14.00
0.60	**64.04**	25.55	10.41
0.70	**58.81**	33.72	7.47
0.80	**52.81**	42.39	4.80
0.90	**44.64**	52.54	2.82
1.0	**1.09**	98.91	0.00

Ensemble			
σ	**Correct**	**Rejected**	**Incorrect**
0	**79.34**	0.00	20.66
0.10	**79.54**	0.00	20.46
0.20	**79.35**	0.00	20.65
0.30	**79.53**	0.02	20.45
0.40	**79.24**	1.05	19.71
0.50	**77.14**	6.73	16.13
0.60	**75.31**	11.23	13.46
0.70	**68.79**	22.62	8.59
0.80	**65.38**	28.00	6.62
0.90	**56.57**	40.23	3.20
1.0	**39.21**	59.83	0.96

RMAggNet $[16, 5, 8]_2$			
EC	**Correct**	**Rejected**	**Incorrect**
3	**77.11**	12.76	10.13
2	**68.32**	27.23	4.45
1	**57.90**	40.09	2.01
0	**42.46**	59.96	0.58

The performance on the CIFAR-10 datasets using open-box attacks is shown in Tables 6 and 7. We, again, aim to minimise incorrectness.

The PGD L_∞ results in Table 6 show low correctness for both Ensemble and RMAggNet for all ϵ which indicates that this is a strong adversarial attack, which is reduced to unrecognisable images at $\epsilon = 0.3$. With this in mind, it is better to compare the methods focusing on incorrectness and rejection performance. Ensemble struggles to reject inputs, leading to nearly 100% incorrectness across all ϵ which indicates that the adversarial examples are able to fool the multiple networks that form this method. RMAggNet shows slightly better performance, with much higher rejection for $EC = 0$. CCAT can achieve the lowest incorrectness scores, which are significantly lower for $\epsilon = 0.75$ and $\epsilon = 2.5$. This indicates that when we expect strong adversaries with little chance of recovery, CCAT is the best-performing model.

Table 7 shows the results for PGD L_2 on CIFAR-10. Interestingly, RMAggNet can outperform both Ensemble and CCAT at $\epsilon = 0.30$ with a lower incorrectness and higher correctness than both methods. For $\epsilon = 0.75$ and $\epsilon = 2.5$ CCAT can report the lowest incorrectness by a significant margin. Over all ϵ values RMAggNet reports lower incorrectness than Ensemble, achieving higher correctness at $\epsilon = \{0.30, 0.75\}$.

These results show the effect that strong adversaries have on the classification ability of these models. In this circumstance, CCAT is the better model, rejecting most adversaries, while Ensemble struggles to reject the inputs, and RMAggNet has varying performance when attempting to correct the images. However, this is a worst-case scenario.

6 Discussion

The results from Section 5 allow us to determine how RMAggNet can be used, and when it may have advantages over competing methods.

We start by discussing the hyperparameters of RMAggNet (see extended paper [17], Section 5.1). The selection of hyperparameters is dependent on the

Table 5: Results for the transfer attacks using PGD L_∞. Table 5a shows the accuracy of the surrogate model on the PGD L_∞ adversarial datasets. Table 5b shows the results of the adversarial datasets on the CCAT, Ensemble and RMAggNet models.

(a) Accuracy of the surrogate CIFAR-10 classifier on the adversarial datasets generated using PGD L_∞ with different perturbation budgets (ϵ).

ϵ	**Accuracy (%)**
0.00	81.21
0.05	0.10
0.10	0.10
0.30	0.00

(b) Percentage of correct, rejected and incorrect classifications of the models using transfer attacks on a surrogate CIFAR-10 classifier using the PGD L_∞ attack. Lower incorrectness is better.

PGD(L_∞)												
CCAT												
τ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	14.80	0.00	**85.20**	0.10	9.50	0.00	**90.50**	0.30	11.20	0.00	**88.80**
0.30		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
0.70		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
1.00		0.00	100.00	**0.00**		0.00	100.00	**0.00**		0.00	100.00	**0.00**
Ensemble												
σ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	57.30	0.00	**42.70**	0.10	37.70	0.00	**62.30**	0.30	16.60	0.00	**83.40**
0.30		56.80	0.00	**43.20**		38.50	0.10	**61.40**		16.90	0.00	**83.10**
0.70		42.30	32.70	**25.00**		26.80	29.50	**43.70**		7.90	38.20	**53.90**
1.00		16.80	77.90	**5.30**		10.00	75.60	**14.40**		0.50	83.80	**15.70**
RMAggNet												
EC	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
3	0.05	44.90	15.30	**39.80**	0.10	21.70	12.90	**65.40**	0.30	2.10	5.20	**92.70**
2		34.20	39.20	**26.60**		16.10	29.70	**54.20**		1.10	11.10	**87.80**
1		24.80	58.30	**16.90**		9.30	48.30	**42.40**		0.50	23.20	**76.30**
0		16.20	75.60	**8.20**		4.40	70.10	**25.50**		0.30	49.40	**50.30**

Table 6: Results of PGD L_∞ adversaries generated using open-box attacks on CIFAR-10 images with percentages of correct, rejected and incorrect classifications. Lower incorrectness is better.

PGD(L_∞)												
CCAT												
τ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	9.00	0.00	**91.00**	0.10	7.10	0.00	**92.90**	0.30	5.40	0.00	**94.60**
0.30		0.00	99.00	**1.00**		0.00	97.50	**2.50**		0.20	83.00	**16.80**
0.70		0.00	99.50	**0.50**		0.00	98.50	**1.50**		0.00	86.40	**13.60**
0.90		0.00	99.70	**0.30**		0.00	98.80	**1.20**		0.00	89.50	**10.50**
Ensemble												
σ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.05	1.10	0.00	**98.90**	0.10	0.00	0.00	**100.00**	0.30	0.00	0.00	**100.00**
0.30		0.90	0.00	**99.10**		0.00	0.00	**100.00**		0.00	0.00	**100.00**
0.70		0.50	2.70	**96.80**		0.00	0.10	**99.90**		0.00	0.00	**100.00**
1.00		0.20	14.80	**85.00**		0.00	1.60	**98.40**		0.00	0.10	**99.90**
RMAggNet												
EC	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
3	0.05	18.50	12.90	**68.60**	0.10	6.40	9.50	**84.10**	0.30	0.20	11.10	**88.70**
2		15.60	27.20	**57.20**		5.00	24.10	**70.90**		0.20	26.20	**73.60**
1		12.80	43.70	**43.50**		4.00	44.90	**51.10**		0.00	50.80	**49.20**
0		9.50	65.20	**25.30**		3.10	67.60	**29.30**		0.00	82.30	**17.70**

Table 7: Results of PGD L_2 adversaries generated using open-box attacks on CIFAR-10 images with percentages of correct, rejected and incorrect classifications. Lower incorrectness is better.

PGD(L_2)												
CCAT												
τ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.30	29.20	0.00	**70.80**	0.75	12.50	0.00	**87.50**	2.5	10.20	0.00	**89.80**
0.30		0.50	92.10	**7.40**		0.00	97.50	**2.50**		0.00	99.50	**0.50**
0.70		0.00	96.00	**4.00**		0.00	99.20	**0.80**		0.00	99.80	**0.20**
0.90		0.00	97.40	**2.60**		0.00	99.70	**0.30**		0.00	99.80	**0.20**
Ensemble												
σ	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
0.00	0.30	54.30	0.00	**45.70**	0.75	26.70	0.00	**73.30**	2.5	13.40	0.00	**86.60**
0.30		53.30	0.00	**46.70**		26.50	0.00	**73.50**		13.30	0.00	**86.70**
0.70		42.00	28.60	**29.40**		23.30	12.90	**63.80**		13.00	0.90	**86.10**
1.00		23.10	70.20	**6.70**		16.20	54.70	**29.10**		12.20	6.30	**81.50**
RMAggNet												
EC	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**	ϵ	**Correct**	**Rejected**	**Incorrect**
3	0.30	64.00	15.30	**20.70**	0.75	39.80	17.00	**43.20**	2.5	11.10	10.80	**78.10**
2		52.40	36.30	**11.30**		32.90	37.90	**29.20**		9.60	23.20	**67.20**
1		41.20	52.10	**6.70**		25.60	54.90	**19.50**		8.20	41.80	**50.00**
0		28.70	69.10	**2.20**		19.40	71.70	**8.90**		5.60	63.00	**31.40**

problem, with the most important aspect being the number of classes the dataset has. If a dataset has $|C|$ classes, then we require at least $\lceil \log_2 |C| \rceil$ networks to generate a unique encoding for each class. This works well for datasets such as MNIST or CIFAR-10, with ten classes each, which requires at least four networks, however, this approach is less optimal for datasets with a small number of classes. For datasets with few classes we are only able to produce $\binom{|C|}{s}$ unique networks (where s is the number of classes we allow in the partitioned set) which limits the number of networks we can use and increases the probability that a random noise input will be assigned a valid class, which decreases adversarial defence.

When deciding on the number of networks to aggregate over (n), we have two constraints. The number of networks must be a power of 2, and we must ensure that $|C| \leq 2^k$, (i.e. we have enough class codewords for the number of classes in the dataset). Since $n = 2^m$, and the value of k is influenced by both m and r, we must balance n with the amount of error correction we need and the probability of a random noise input being assigned a valid class. The results presented in this paper have focused on datasets with 10 and 47 classes, requiring 16 and 32 networks respectively. If we consider extending this approach to ImageNet with 1000 classes, we can see the effect of scaling on RMAggNet. For 1000 classes we have the restriction that $1000 \leq 2^k$, therefore $k \geq 10$. If we use 32 networks ($m = 5$, $r = 2$) we get $k = 16$ with 3 errors corrected ($t = 3$) with a probability of a random noise input being assigned a valid class of 1.28×10^{-3}. This indicates that scaling to datasets with more classes is feasible using RMAggNet.

The comparisons between RMAggNet, Ensemble and CCAT over the EMNIST and CIFAR-10 datasets on clean and adversarial inputs allow us to place RMAggNet in context with the other methods. The results for these tests are in sections 5.1, and 5.2 (more results available in the extended paper [17]).

Results on the clean testing data (Tables 1 and 4) show that RMAggNet is able to train models which are competitive with the other architectures, equalling Ensemble for some datasets. This result shows that the RMAggNet method has minimal impact on clean dataset performance.

Across the adversarial tests CCAT is able to reject the most adversarial inputs leading to nearly 0 incorrect classifications. However, CCAT does not attempt correction of any inputs which leads to 0 correct classifications at most confidence thresholds. This even occurs when both Ensemble and RMAggNet recover over 80% of the labels from an adversarial attack (Table 3). This approach to rejection means that CCAT is ideal for situations where any uncertainty in the correctness of a result cannot be tolerated. However, if we can allow some incorrectness, and want the option to reject, then Ensemble and RMAggNet allow us to classify many inputs correctly, greatly reducing the reliance on downstream rejection handling, at the risk of small amounts of incorrectness. If we compare the Ensemble method with RMAggNet, over many of the datasets RMAggNet is able to outperform Ensemble with a slightly higher (or equal) number of correct classifications, and lower incorrectness for comparable correctness as it uses the reject option more effectively. This becomes more pronounced at higher ϵ. The application of RMAggNet to datasets with many more classes, such as ImageNet, would be interesting future work since we have stated that $|C| \leq 2^k$ (section 3.2), and this can be achieved by either adding more networks (increasing m) or increasing the polynomial degree (r) which decreases error correction ability and increases the probability of assigning random noise a class. Striking this balance would lead to interesting results regarding the applicability of RMAggNet to larger datasets.

7 Conclusion

In this paper we have seen how an architecture leveraging Reed-Muller codes can be used to create an effective CWR method which (to our knowledge) is the first approach combining its rejection and correction ability. The experimental results show the advantages of RMAggNet and allow us to determine situations where it can be beneficial to a system. Comparing the results of RMAggNet to CCAT, shows that CCAT is able to reject nearly 100% of the adversarial images over all attacks and datasets we tested. However, this comes at the cost of rejecting inputs that could otherwise be classified correctly. The sensitive approach of CCAT could be detrimental to a system where the rejected inputs still need to be processed, either using more computationally expensive processes, or reviewed by a human. RMAggNet is able to achieve low incorrectness, often with higher correctness, meaning that, provided the system can accept small amounts of incorrectness, we can reduce the reliance on downstream rejection handling. From the results, we can see that this can be a significant improvement for small perturbations on the EMNIST dataset. Comparing the results of RMAggNet to Ensemble, the results show that RMAggNet appears to be more resilient to adversarial attacks, particularly open-box attacks, often achieving lower incorrectness than Ensemble by having higher correctness for small ϵ, or higher rejection for larger ϵ. This means that we can expect RMAggNet to be a better choice in situations where adversaries are present.

References

1. X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh *et al.*, "Symbolic discovery of optimization algorithms," *arXiv preprint arXiv:2302.06675*, 2023.
2. A. Tragakis, C. Kaul, R. Murray-Smith, and D. Husmeier, "The fully convolutional transformer for medical image segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 3660–3669.
3. F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro, "Intriguing properties of adversarial ml attacks in the problem space," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 1332–1349.
4. C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv:1312.6199*, 2013.
5. L. Smith and Y. Gal, "Understanding measures of uncertainty for adversarial example detection," in *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, vol. 2, mar 2018, pp. 560–569.

6. A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023.
7. J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 119–126.
8. S.-T. Chen, C. Cornelius, J. Martin, and D. H. Chau, *ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector*. Springer International Publishing, 2019, p. 52–68. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-10925-7_4
9. N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 582–597.
10. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
11. G. Verma and A. Swami, "Error correcting output codes improve probability estimation and adversarial robustness of deep neural networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
12. C. Cortes, G. DeSalvo, and M. Mohri, "Learning with rejection," in *Algorithmic Learning Theory: 27th International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings 27*. Springer, 2016, pp. 67–82.
13. N. Charoenphakdee, Z. Cui, Y. Zhang, and M. Sugiyama, "Classification with rejection based on cost-sensitive classification," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1507–1517.
14. Y. Song, Q. Kang, and W. P. Tay, "Error-correcting output codes with ensemble diversity for robust learning in neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9722–9729.
15. N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.
16. D. Stutz, M. Hein, and B. Schiele, "Confidence-calibrated adversarial training: Generalizing to unseen attacks," *Proceedings of the International Conference on Machine Learning ICML*, 2020.
17. D. Fentham, D. Parker, and M. Ryan, "Using Reed-Muller codes for classification with rejection and recovery," *arXiv preprint arXiv:2309.06359*, 2023.
18. A. Gamal, L. Hemachandra, I. Shperling, and V. Wei, "Using simulated annealing to design good codes," *IEEE Transactions on Information Theory*, vol. 33, no. 1, pp. 116–123, 1987.
19. D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," *Transactions of the I.R.E. Professional Group on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, 1954.
20. I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954.
21. R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.
22. N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

23. A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” *arXiv preprint arXiv:1706.06083*, 2017.
24. W. Brendel, J. Rauber, and M. Bethge, “Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,” in *International Conference on Learning Representations*, 2018.
25. G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, “Emnist: Extending mnist to handwritten letters,” in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.
26. J. Rauber, R. Zimmermann, M. Bethge, and W. Brendel, “Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax,” *Journal of Open Source Software*, vol. 5, no. 53, p. 2607, 2020. [Online]. Available: https://doi.org/10.21105/joss.02607
27. J. Rauber, W. Brendel, and M. Bethge, “Foolbox: A python toolbox to benchmark the robustness of machine learning models,” in *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017.
28. A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in *International conference on machine learning*. PMLR, 2018, pp. 274–283.
29. Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting adversarial attacks with momentum,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.
30. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
31. P. Jeevan, K. Viswanathan, and A. Sethi, “Wavemix-lite: A resource-efficient neural network for image analysis,” *arXiv preprint arXiv:2205.14375*, 2022.