Investigating Optimal Constant Solutions in Deep Neural Networks

Anton Adelöw
aadelow@kth.se

Santiago Bou Betran
sbb@kth.se

Jayesh S Vasudeva
jayeshv@kth.se

Abstract

To deploy neural networks for decision critical applications we need metrics or protocols to map the reliance of an input-output pair for a neural network. This report highlights a trend in the extrapolated outputs of the deep neural network called **Optimal Constant Solution**[7]. The chosen paper, "Deep Neural Networks Tend To Extrapolate Predictably,"[7] aims to dissect the behavior of neural networks when exposed to data originating from a distinct distribution, showcasing its implications for classification and prediction certainty. The central premise revolves around the observation that, under the influence of high-dimensional input, the model's output tends to converge towards an Optimal Constant Solution (OCS) We replicate the results of Kang et al.[7] with Resnet-20 backbone for CIFAR-10 dataset. Moreover, we study the effect of two more Vision-transformer based models CoatNet[1] and CaiT[10]. We also share our findings when trained under an adversarial setting, as well as explore the effect of weight decay on the reversion to the OCS. In addition, we experiment with the multi margin loss function to see how the choice of loss function affects the reversion to the OCS. We get stronger results reversion to the OCS for CoatNet and Cait than for ResNet20, and find that the choice of loss function and regularization strength clearly affect the reversion to the OCS.

1 Introduction

We study how neural networks with high-dimensional inputs extrapolate predictably under out-of-distribution (OOD) data, following the paper *'Deep Neural Networks Tend To Extrapolate Predictably'*[7]. The paper shows that neural networks can classify and predict OOD data with some certainty, as their output converges to the Optimal Constant Solution (OCS) when the input gets further from the original training dataset. The paper provides both theoretical and empirical evidence for this phenomenon. The theoretical background is based on analyzing a simplified network with ReLU activation and various datasets with different levels of corruption. From the empirical perspective, the paper tests the possibility of using the distance from the output to the OCS can be used as a measure of uncertainty or risk for the network's prediction, and that a result can be rejected if it is too close to the OCS.

We replicate the paper's empirical experiments, using the same network architecture and the CIFAR10 and CIFAR100 datasets. On top of the replication of these main results, we extend our study by comparing the performance of two other networks, CaiT transformers and CoatNet-based architecture, on the same tasks. Moreover, we use different uncertainty range metrics to evaluate the results with different seeds, and we test an adversarial network model to see the effect on the output towards reversion to OCS. Code at: `https://gits-15.sys.kth.se/sbb/DLAdv-Project-Extrapolation`

2 Related Work

The paper 'Deep Neural Networks tend to Extrapolate Predictably' by Kang et al. [7] elucidates a factor from [6] that displays a difference in trends amongst in-distribution data(which is hard to map) and the out-of-distribution(OOD) data which can be stated as less confident predictions were observed for OOD when compared. Kang et al. [7] puts further effort into quantification and qualification of the via Optimal Control Solution and the extent of the sample belongingness in-distribution.

We employ Dai et al.[1] and Trouvron et al.[10] architectures to further the study the cause on a broader family of models. Both models have a transformer based approach to representations and slightly alter the model to provide state-of-the-art performance on classification with Imagenet, CIFAR, etc.

3 Methods

The original paper describes different scenarios to prove that the appearance of OOD data in an input makes the model converge towards an OCS-like family of results. To prove this claim, the paper performs several empirical explorations with different datasets (CIFAR10, ImageNet, DomainBed, SkinLesionPixels, UTKFace, BREEDS and WILDS). For each of these datasets, several noise types and distortions are introduced in different levels which are then used to evaluate the results of a classifier. For this classifier, several exploration techniques are done, such as extracting the values of the parameters in different layers and performing comparisons between the effect of using several loss functions with the usage of more OOD data in the inputs. Some of the loss functions used cross entropy (CE), mean squared error (MSE), and regression with Gaussian NLL.

3.1 Model Architectures

3.1.1 ResNet20

The first architecture we implemented is based on the paper and corresponds to the well known ResNet20 Residual Network model presented by Kaiming He in [3]. For brevity, we refer to the Figure 1 for its implementation, highlighting the Residual connections which allow to train deeper models.

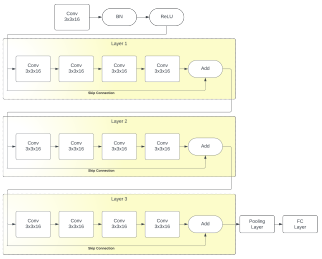

Figure 1: Architecture of ResNet20

Figure 2: Hyper-parameters used for ResNet20

learning rate	0.001
scheduler	Decaying LR
optimizer	Stochastic Gradient Descent
weight decay	0.0001
momentum	0.9
batch size	128
epochs	200
loss function	Cross-Entropy

One of the main advantages of these types of residual networks, which makes it also interesting for the proposed experiments, is the use of skip connection in the different layers. With these connections, we allow the network to connect the input signal of each block to the output and as such avoid problems such as vanishing or exploding gradients that arise from the use of deep architectures. This effect can also be observed with the reversion to the OCS principle, which is further increased in the presence of this type of connection.

3.1.2 CoatNet

The second model we implmented is based on the CoAtNet architecture described by Dai et al. in [1]. Essentially, CoAtNet is a hybrid model combining depthwise convolution and self-attention by vertically stacking the said layers. More formally, the CoAtNet architecture consists of five stages, with gradually decreasing spatial resolution. The first stage consists of a 2-layer convolutional

block. The next two stages consists of MBConv blocks [9], utilizing depthwise convolution and squeeze-excitation. The last two stages are Transformer blocks, employing multi-head attention with relative bias, and a feed forward network. The model includes connections in between all layers, as well as inside the transformer blocks in between the attention mechanism and the feed forward network. This is done by using the pre-activation [4]. Lastly global pooling is performed, and a fully connected layer then maps the output. The GELU activation function [5] is used throughout the architecture. At each stage, the spatial size is reduced and the number of channels increased. In [1], the down-sampling in each block results in decreasing the resolution by half each time. Since the images in CIFAR-10 and CIFAR-10 are only 32×32, this does not make sense in our case. Therefor, we use less aggressive downsampling as a modification of the architecture, where we only downsample once in stage 3. The architecture can be seen in Figure 3.

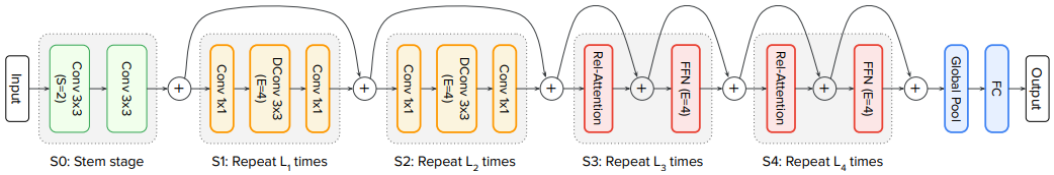

Figure 3: Architecture of CoAtNet [1]

The specific architecture of our CoAtNet implementation can be seen in Table 3, where L denotes the number of blocks in each stage, and D denotes the hidden dimension (number of channels). For more details about the CoAtNet architecture, we refer to the original paper [1].

Table 1: CoAtNet Architecture Parameters.

Stages	Size	CoAtNet	
S0-Conv	1	L=1	D=64
S1-MbConv	1	L=1	D=64
S2-MBConv	1/2	L=2	D=128
S3-TFM	1/4	L=2	D=256
S4-TFM	1/16	L=2	D=512

Table 2: Hyper-parameters used for CoAtNet

learning rate	0.001
scheduler	OneCycleLR
optimizer	AdamW
weight decay	0.1
momentum	0.9
batch size	128
epochs	100
loss function	Cross-Entropy

3.1.3 CaiT Transformer

The third model we implemented is based on the study of deep transformer architectures by Touvron et al.[10]. The architecture highlights that depth doesn't bring meaningful improvement for the model even though the residual connections are meant to preserve the features learnt throughout the depth of the model. To conquer these points, the architecture proposes a delayed injection of `CLS` token in the model and propose `LayerScale` as an added measure in the residual connection within the transformer block.

A transformer block when imagined as a set of residual connections can be written as:

$$\begin{aligned} x'_l &= x_l + SA(\eta(x_l)) \\ x_{l+1} &= x'_l + FFN(\eta(x'_l)) \end{aligned} \tag{1}$$

where η is `LayerNorm`, SA and FFN are Soft-attention and Feed-forward Network for the input x_l for the l^{th} block. Touvron et al.[10] proposes to multiply a learnable diagonal matrix with the output of soft-attention and feed-forward layer to facilitate the convergence and improve accuracy. The revised transformer block equation can now be written as:

$$\begin{aligned} x'_l &= x_l + diag(\lambda_{l,1}, ...\lambda_{l,d}) \times SA(\eta(x_l)) \\ x_{l+1} &= x'_l + diag(\lambda'_{l,1}, ...\lambda'_{l,d}) \times FFN(\eta(x'_l)) \end{aligned} \tag{2}$$

where parameters $\lambda'_{l,i}$ and $\lambda_{l,i}$ are learnable parameters with initialization close to 0 depending on the depth of the block.

Table 3: CaiT Architecture Parameters

Params	Value
Soft-Attention-depth	5
Class-Transformer-depth	2
Embedding-dim	358
#Heads	6
Mlp-dim	240

Table 4: Hyper-parameters used for CaiT

Maximum LR(during warmup)	5e-4
LR scheduler(post warmup)	Exponential LR
optimizer	AdamW
weight decay	0.1
Layer Dropout	0.05
Embedding Dropout	0.1
batch size	128
Warmup epochs	150
epochs	500
loss function	Cross-Entropy

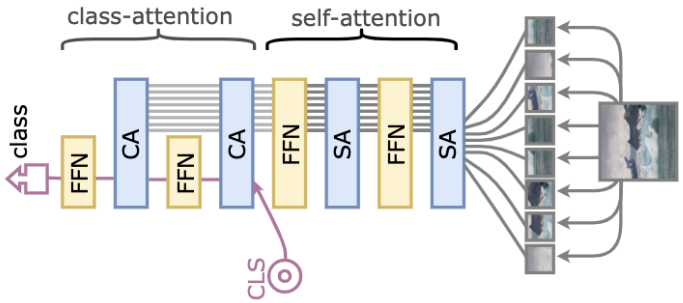

Figure 4: CaiT model design[10]

As illustrated in Fig. 4 the self-attention is similar to ViT except `CLS` token not being a part of the self-attention process. Instead, it's part of a different segment of the architecture called 'class-attention'. Class attention is responsible for collecting the patches from self-attention with `CLS` token to make a feed it to the classifier.

3.2 Loss Functions

In addition to training the models with cross-entropy loss, we have also used multi margin loss to fit the models. This is done to study how the choice of loss function affects the reversion to the optimal constant solution. We apply softmax to model outputs before multi margin loss, which results in the optimal constant solution for the multi margin loss function being simply being the set of probabilities that match the empirical distribution of the labels in the training set.

3.3 Regularization Methods

Our models are trained using Stochastic Gradient Descent optimizer for the case of the ResNet20 model and AdamW for CoatNet and CaiT models. In order to study the effect of regularization methods on the reversion to the OCS, we vary the weight decay and dropout parameters for the models. We expect the regularization to make the outputs show a more stable and reliable approach towards the OCS when presented with more noisy data, as the model is able to learn better the higher order features of the input data.

3.4 Adversarial Training

To analyze how adversarial training affects the reversion to the OCS, the fast gradient sign method [2] was implemented. For each example in each batch, an adversarial example is created and trained on, where we used the value $\epsilon = 0.01$. Usint these training technique, we expect to be able to result whether the model is better able to learn the data, and as such, better recognize when an input does not belong to the original distribution, as it happens during the training when incorporating adversarial examples.

4 Data

In this project, we are centring our efforts on analyzing two of the most prominent datasets in the image recognition field, corresponding to the CIFAR10 and CIFAR100 datasets defined by Alex Krizhevsky [8]. We can concisely describe it as a set of 60.000 images divided in 10 and 100 classes respectively with a resolution of 32x32. We use a test set of 10.000 out of those images for our experiments. To improve the learning diversity and generalization, and as done in the paper. We use a set of transformations based on PyTorch primitives: RandomHorizontalFlip (mirror of the images along the vertical axis, with a 50% probability), RandomCrop (crop subsections of size 32 and padding of 4 around the edges of the image), Normalization (subtract the mean and dividing the results by the standard deviation of the images)

5 Experiments and findings

To check the initial premise for the reversion to the OCS, we followed the described testing methodology explained in the original paper. In our case, the implementation of the networks is done using PyTorch 1.13.1. The used hyper-parameters are presented in Tables 2, 4, 2.

With the trained model, we study the effect of out-of-distribution data as input on the model. We use a corrupted version of the CIFAR10 dataset named CIFAR10-C, which offers 5 different levels of corruption of the input images with 5 different types of noise, corresponding to Impulse Noise, Shot Noise, Defocus Blur, Motion Blur and Speckle Noise. The experiments analyze the network outputs for each type of corruption for the test set. Afterwards, we calculate the distance from the obtained outputs to the OCS solution, which we define as the average target value for the MSE metric or the prior for each class for the Cross Entropy. We plot the distance to the OCS against the Score of the model's solution.

We define the score of an output as the relative difference between the original data and the corrupted OOD sampled images. To compute this metric we used a simple one-layer linear network with a sigmoid function which we train for 30 epochs to distinguish a subset of the training and corrupted data. We obtain the softmax probabilities representing for each datapoint in the corrupted dataset how close it is to an in-distribution sample. We expect lower corruption levels to be placed closer to the original dataset, remarked as "Oracle" in the results. We can observe the resulting plot using cross entropy and MSE loss in Figure 5 and 6.

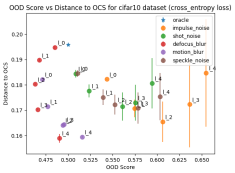

Figure 5: CE score

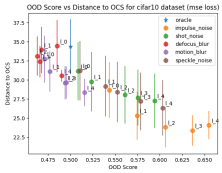

Figure 6: MSE Score

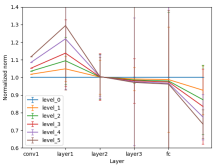

Figure 7: Layer Norm

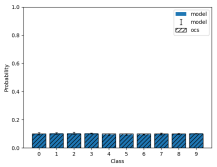

Figure 8: OCS similarity

Plots for ResNet20 architecture on using CIFAR10-C test dataset against OCS metrics

From Figure 5 and 6, the obtained results support the thesis of the reversion to OOD. We can observe a linear trend, where those datasets with a higher OOD score, result in a more similar output (closer distance) to the OCS. We can see that this effect is clearer for the MSE Loss function and the use of impulse, speckle or shot noise. On the other side, the use of blurring as a way of obtaining OOD data does not result in as good scores as the other models, but we can still observe how higher corruption levels lead to lower distances to the OCS.

To conclude the replication experiment on the paper results, we show the effect of using these different levels of corrupted data in the value of the parameters of the model. For this, we compute the normalized norms of the parameters in each of the different layers to evaluate the evolution of its values. We use for this the "impulse noise" corrupted dataset.

As well, we can observe how the mean of the class probabilities, for a constant input consisting of zeros to the trained model, also closely resembles the expected OCS solution, which assigns an equal probability to each class. We can observe the results of these two experiments in Figures 7, 8.

Figure 5 shows how the parameters diverge more from the normalized solution as the corruption level becomes higher. We can also observe how the standard deviation becomes broader for those increased levels, meaning that OOD samples will affect greatly the parameter values on each of the layers of the network. We can also observe in Figure 8 how a complete OOD input such as a vector of zeros, will, as expected, result in a uniform OCS output from the trained model with cross-entropy loss. In figures 10 and 9, we see the results of the CoAtNet and Cait models. Here we observe a clear reversions to OCS when the noice levels are increasing. The model architecture seems to have a strong effect on the observed reversion to OCS, which is not suprising. For CoAtNet trained on CIFAR-100, we do not observe this. We speculate the reason for this is that the model is fitted worse to this dataset and thus does not extrapolate predictably. CoAtNet trained on CIFAR-100 with reward loss seems slightly better than with cross-entropy loss, however.

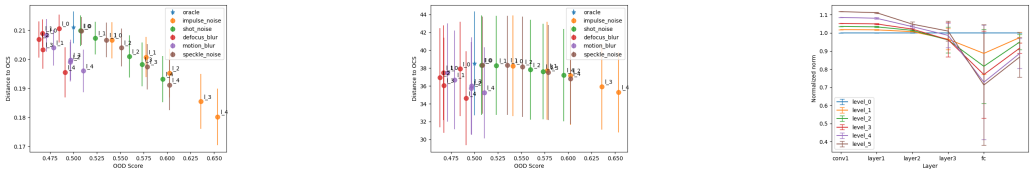

Figure 9: Results of training CaiT on CIFAR-10. From left we show the score using Cross Entropy and MSE. The last plot shows the normalized parameter values along the layers of the model. (Error bars are shown for the standard deviation among training 3 seeds)

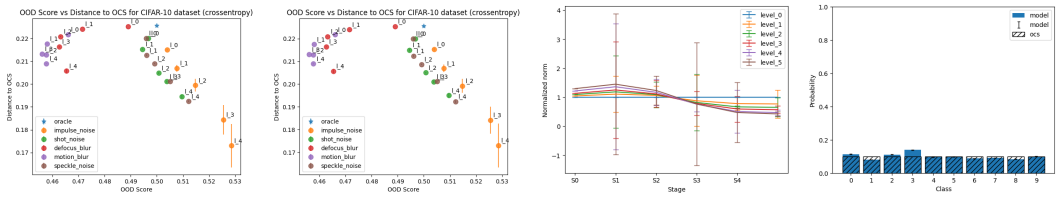

Figure 10: Results of training CoAtNet on CIFAR-10. From left we show the score using Cross Entropy and MSE. The third plot shows the normalized parameter values along the layers of the model and the last plot shows OCS similarity when the last layer recieves the zero vector as input. (Error bars are shown for the standard deviation among training 3 seeds)

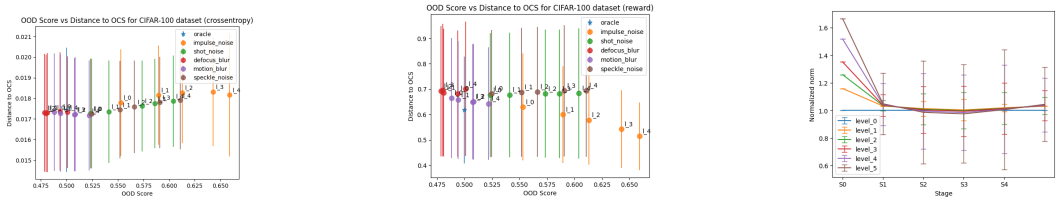

Figure 11: Results of training CoAtNet on CIFAR-100. From left we show the score using Cross Entropy and MSE. The last plot shows the normalized parameter values along the layers of the model. (Error bars are shown for the standard deviation among training 3 seeds)

5.1 Changing the Loss Function: MML

The first extension we experiment over the original paper is the use of a different Loss function for training.

In Fig 12 we can observe how the use of the MML function actually reduces the effect of the reversion to the OCS solution. We can see how the increase in noise levels on the input does not equate in a linear reversion to the OCS solution, as we saw in the previous case. This means that the model is performing a more generalized learning strategy and as such it gives a too confident solution when presented with OOD samples. For CoAtNet, in Fig 13 we observe a different trend, where increasing

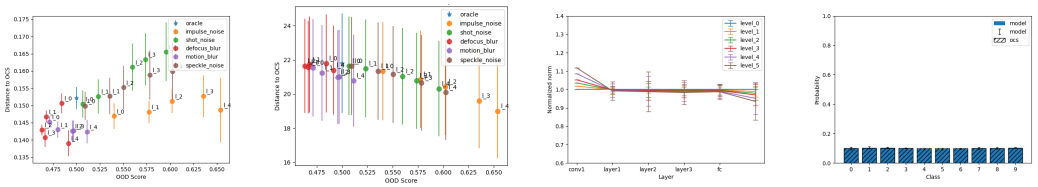

Figure 12: Results of training ResNet20 with MML on CIFAR-10. Starting by the leftmost plot, we show the score using Cross Entropy and the next one plots using MSE. The third plot shows the normalized parameter values along the layers of the model and the last is the class result similarity for a zero constant input against the OCS. (Error bars are shown for the standard deviation among training 3 seeds)

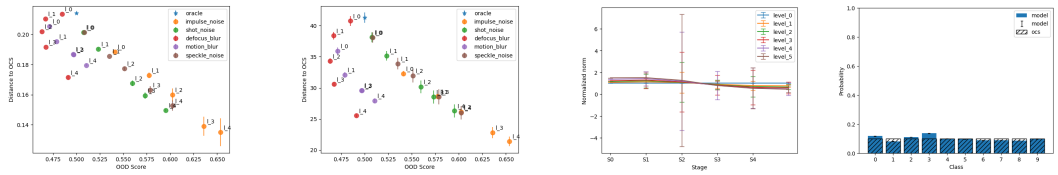

Figure 13: Results of training CoAtNet with MML on CIFAR-10. Starting by the leftmost plot, we show the score using Cross Entropy and the next one plots using MSE. The third plot shows the normalized parameter values along the layers of the model and the last is the class result similarity for a zero constant input against the OCS. (Error bars are shown for the standard deviation among training 3 seeds)

levels of noice seem to correspond to a reversion to the OCS, indicating a relationship between the loss function and model architecture with regards to the reversion to the OCS.

5.2 Using Adversarial Training

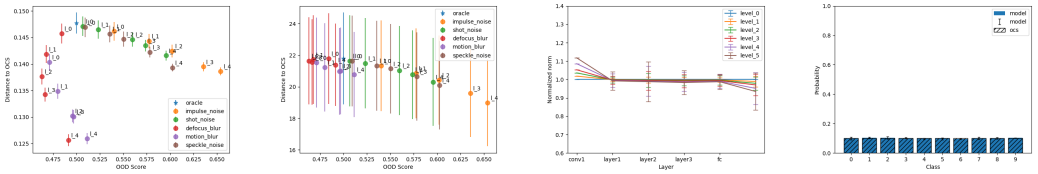

Figure 14: Results of training ResNet20 on CIFAR-10 with Adversarial Training. Starting by the leftmost plot, we show the score using Cross Entropy and the next one plots using MSE. The third plot shows the normalized parameter values along the layers of the model and the last is the class result similarity for a zero constant input against the OCS. (Error bars are shown for the standard deviation among training 3 seeds)

We can observe the output of the model when using an adversarial training regime, which results in a more interesting plot. From Fig 14, we can see how the normalized layers show much higher values for the initial and last layers. As well, we can remark that there is a linear trend in the distance as samples become more OOD, especially noticeable for the impulse noise results. We can also comment that the last plot shows an output very close to OCS when using a zero input in the last 3 model layers, showing that the constant parameters of the model are related to a uniform distribution of values. In Fig 15, we observe a very similar trend, indicating that this effect of adversarial training is not model-specific.

5.3 Reward Loss training

Another experiment we replicated and explored in this work is the use of a reward loss model, which as described in the paper, allows the model to choose the option to abstain when presented with OOD

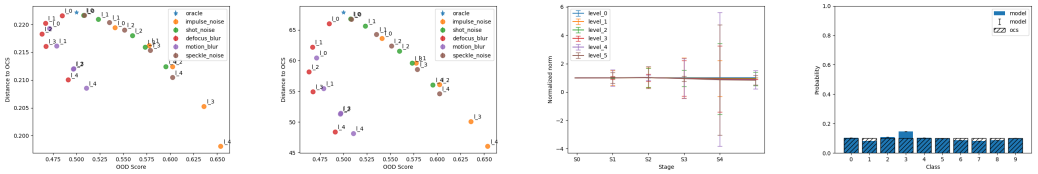

Figure 15: Results of training CoAtNet on CIFAR-10 with Adversarial Training. Starting with the leftmost plot, we show the score using Cross Entropy and the next one plots using MSE. The third plot shows the normalized parameter values along the layers of the model and the last is the class result similarity for a zero constant input against the OCS. (Error bars are shown for the standard deviation among training 3 seeds)

inputs. To train this model, we assign a negative (-4) value for the samples that are not classified correctly, 0 if it abstains from giving an output, and 1 if it gives the correct solution. Based on this model, we analyze the behaviour of the trained networks when presented with data that includes more corruption levels.

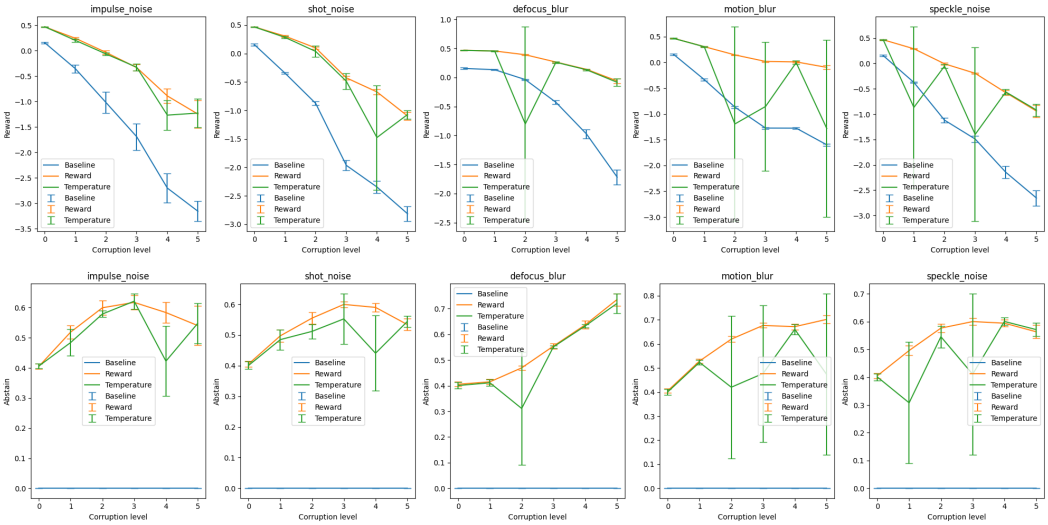

Figure 16: Results of training ResNet20 with Reward Loss on CIFAR-10. On the left, we show the average reward and on the right we plot the ratio of abstention on the Test Set. (Error bars are shown for the standard deviation among training 3 seeds)

As we can observe in Figure 16, when the model is presented with more OOD inputs, the reward decreases and abstaining rate increases. We can observe some interesting results when presenting the model with blurred inputs, which have the greatest effect in the model choosing to abstain from classification.

As well, the implementation of the temperature (Called Oracle model) which should correspond to a higher upper ceiling in which we can base our comparisons for the performance of the models. However, as we explain in the Challenges section, this function is not properly defined and the implementation we followed seems to lead to different results. We think this might be because of our ResNet20 implementation as well as the use of different hyperparameters for the training. For the CoAtNet and Cait models, the temperature was ignored when plotting.

The CoAtNet and Cait models can be seen in figures 17 and 18, respectivly. The plots for the two models look very similar, and we observe a clearer increase in abstain rate and higher rewards. This indicates that these architectures may be better than ResNet20 suited for decision critical applications.

Finally, in Figure 19 we did an extra study of the results for CIFAR-100, we can observe how the reward loss values have decreased significantly compared to the 10 class problem. However, we can still clearly observe how the reward decreases as the corruption level becomes bigger, thus showing

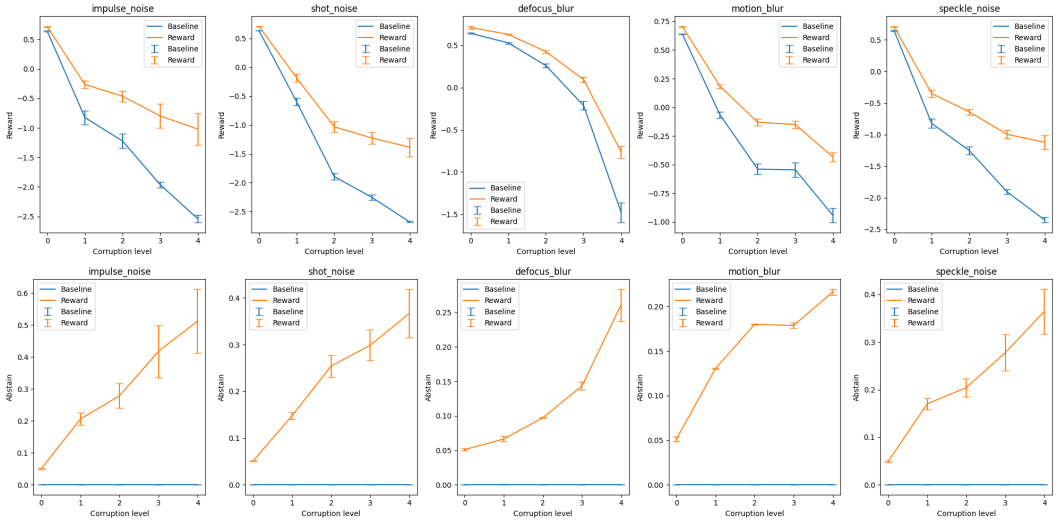

Figure 17: Results of training CoAtNet with Reward Loss on CIFAR-10. On the top, we show the average reward and on the bottom, we plot the ratio of abstention on the Test Set. (Error bars are shown for the standard deviation among training 3 seeds)

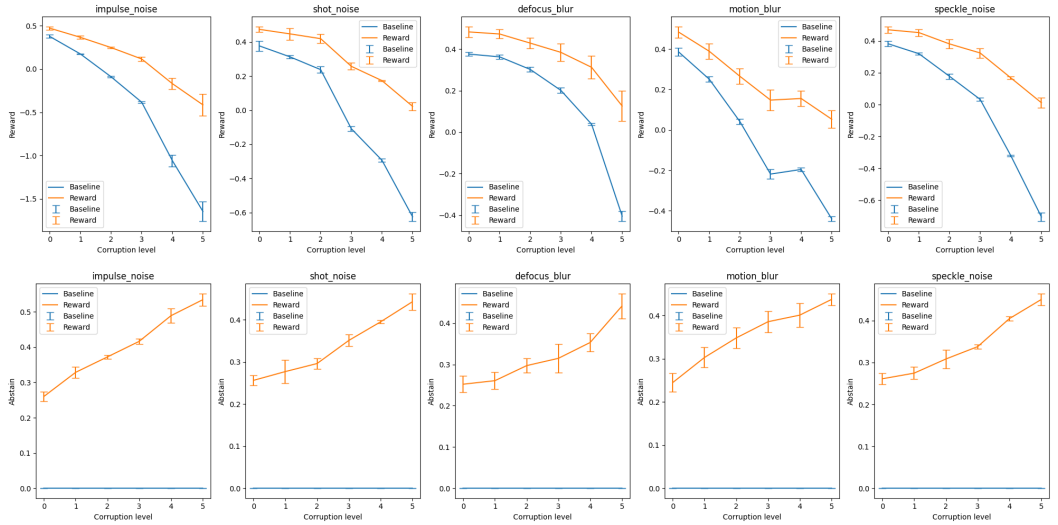

Figure 18: Results of training CaiT with Reward Loss on CIFAR-10. On the top, we show the average reward and on the bottom, we plot the ratio of abstention on the Test Set. (Error bars are shown for the standard deviation among training 3 seeds)

that the model is unable to distinguish noisy data as efficiently as in other experiments such as ResNet20. Observing the abstention rate, we can also see how we can not observe a higher value as we have more OOD data, such as proposing that the model does not have enough capacity with the given training regime to differentiate an output that does not belong to the original set.

5.4 Increasing Regularization

Another hypothesis we tested outside the scope of the original paper is the effect of a bigger regularization term in the training regime of ResNet20. In such a sense, we will modify the original parameter for the Weight Decay function of the SGD regularize to a value of 0.1 increasing from the original 0.0001. With this change we expect the model to be able to better generalize the inputs and as a result, not be able to converge to the OCS for OOD samples.

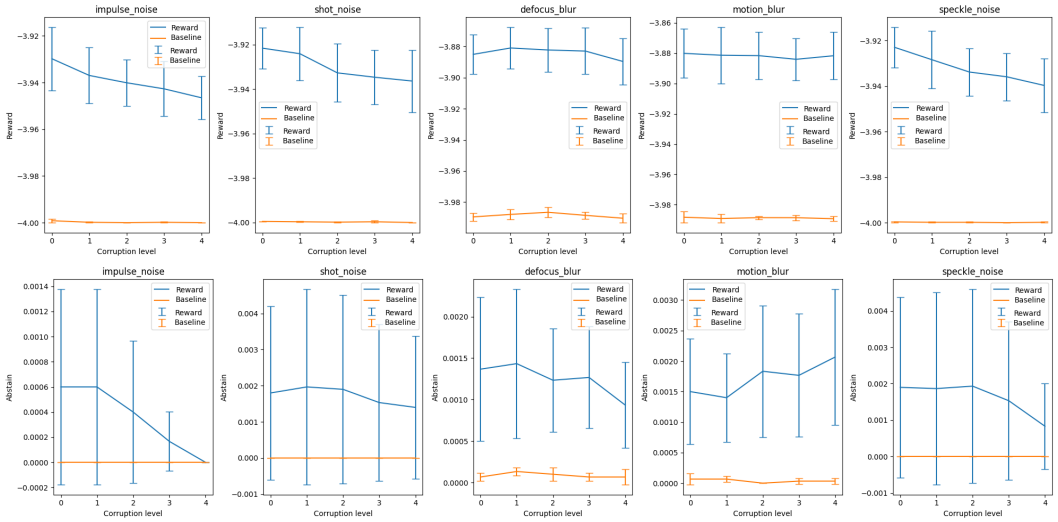

Figure 19: Results of training CoAtNet on CIFAR-100 with Reward Loss. On the top, we show the average reward and on the bottom, we plot the ratio of abstention on the Test Set. (Error bars are shown for the standard deviation among training 3 seeds)

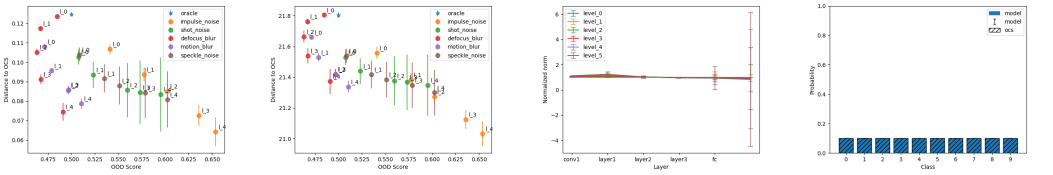

Figure 20: Results of training ResNet20 with Increased Regularization. Starting by the leftmost plot, we show the score using Cross Entropy and the next one plots using MSE. The third plot shows the normalized parameter values along the layers of the model and the last is the class result similarity for a zero constant input against the OCS. (Error bars are shown for the standard deviation among training 3 seeds)

In Figure 20, we can observe how as expected, the values for both types of errors are greatly decreased in average from the previous figures for each type of noise. However, we can still clearly see a linear trend where the inputs become closer to the OCS as they show more OOD features. We can also observe how the normalized values for the layers of the model is also less pronounced, as we can also see, the values of the parameters on the output layer increases as we increase the corruption level, which is a sign the model is less able to obtain a good stable solution.

We can further observe this using the reward model with abstaining in Figure 21, here we see a great difference with the previous model, specially in how the abstaining function now obtains much better results, showing how the model is able to abstain in almost all the cases when the corruption levels are the highest for every type of noise. This further motivates the usefulness of regularization as it helps the model not over-confidently classifying

6 Challenges

Some challenging problems we found were mainly caused by a lack of detail and description of the experiments described on the corpus. We can highlight that the appendix in the original paper serves as a very useful starting point. As such, some of the most troublesome parts are the definition of the OCS metric, explained at a very high level in the paper. Another challenge was implementing the Oracle for the reward function. In the paper it is barely mentioned that they implemented a temperature scaling of the outputs of the model during the testing for computing this ideal value. As a result, we found no other indication of the description of this procedure and after trying our

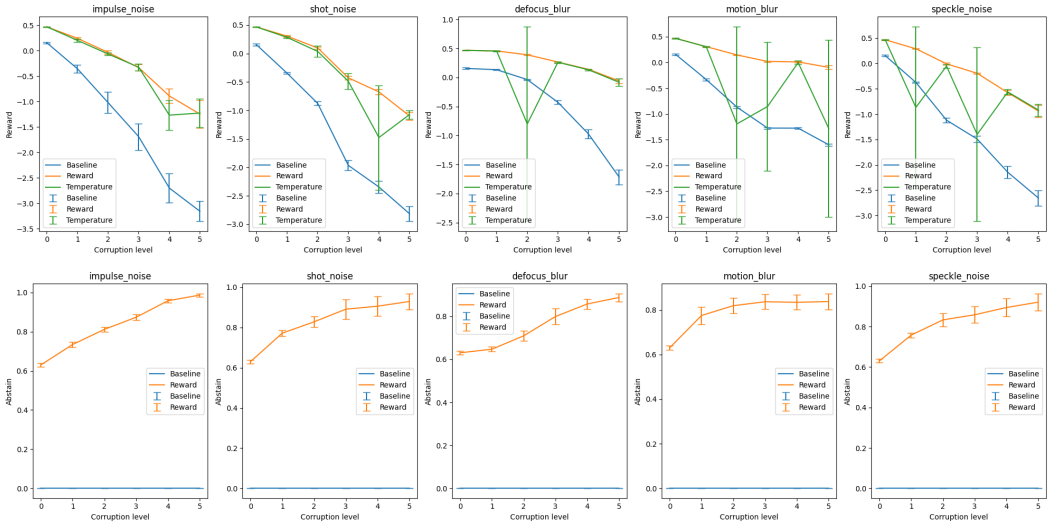

Figure 21: Results of training ResNet20 with Increased Regularization and reward loss.(Error bars are shown for the standard deviation among training 3 seeds). We show also the results of using temperature as an Oracle

implementation and observing the code they provided. We suspect the use of of different seeds can also have a part as we can observe a very high variance when computing this temperature function.

7 Conclusion

From the presented results, we can conclude that the effect of reversion to the OCS, which is the main topic presented in the paper, is replicable and true. However, during our implementations and different experiments, we could find some cases where this reversion was not closely followed, as with the use of blur based noise. Other interesting findings can be observed by the different network models, which also seem to be affected by the Reversion to OCS, as we can see how the inputs become linearly closer to the OCS in every case. We can also observe this when using the reward functions, which also support this thesis of classifiers being able to use this metric to abstain from classifying problematic data.

In such way, we can also argue that some of the experiments and metrics used in the paper, specially with the value computation for the OCS, were not specified clearly enough. As well as the other methods explained in the Challenges section, we can argue that even though the paper provides a good basis on the theoretical understanding, the quality of the provided explanations as well as the implementation provided was very poorly documented and explained in the original work. We do believe that without the access to the original implementation, there would have been parts of the report, specially concerning the last reward function definition and its baseline, that would be impossible to replicate with enough confidence. We also missed some more insights in the work done to tune the hyperparameters as well as the design choices for some of the model architectures, which were not stated in the original paper, such as using Kaiming He weight initialization in ResNet.

8 Ethical consideration, societal impact, alignment with UN SDG targets

The results of such kind of studies as this one can have a great impact on how further studies and developments are carried out. For example, having a metric such as OCS would allow developers to establish a reliable metric for the certainty of the networks, and as such reduce the number of false positives that can be especially important in some fields such as medicine or critical sectors where the safety of the results is of great importance. Also, the use of generic datasets from the web also represents a limitation in the scope of the results, where more work towards ensuring a representation of a broader range of problems and data could lead to different results from the ones obtained. According to the UN SDG targets, we can say it aligns with the goal 4 of establishing quality of education, as it aim to enhance the safety of AI systems and better guide future research in the field.

9 Self Assessment

We assess our report as **Excellent**(B-A) for the following reasons:

- The paper shortlisted for the report is very recent and addresses one of the active research problems.
- The code mentioned in the project report highlights some key steps which the paper[7] seems to miss and we have highlighted those key steps in the report as well as our version of the code.
- We conducted extensive experiments. We conducted experiments acoss 3 different family of models namely Resnet, CoatNet and CaiT. With no prior knowledge of ideal set of hyperparams for the task, given the low compute has been challenging.
- We higlight that we have further refined the research results from the paper. As the complexity of the model increases, the trends observed are coherent with respect to the results observed in Kang et al.[7].

We Would further like to nominate ourselves for the bonus points as we have extended the application to another dataset - CIFAR-100 with CoatNet[1].

References

[1] Z. Dai, H. Liu, Q. V. Le, and M. Tan. Coatnet: Marrying convolution and attention for all data sizes. *arXiv preprint arXiv:2106.04803*, 2021.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[5] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[6] D. Hendrycks, M. Mazeika, and T. Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

[7] K. Kang, A. Setlur, C. Tomlin, and S. Levine. Deep neural networks tend to extrapolate predictably. *arXiv preprint arXiv:2310.00873*, 2023.

[8] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. (0), 2009.

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[10] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 32–42, 2021.