Mitigating Risk in Neural Network Classifiers

Misael Alpizar Santana
Department of Computer Science
University of York, York, UK
misael.alpizarsantana@york.ac.uk

Radu Calinescu
Department of Computer Science
University of York, York, UK
radu.calinescu@york.ac.uk

Colin Paterson
Department of Computer Science
University of York, York, UK
colin.paterson@york.ac.uk

***Abstract*—Deep Neural Network (DNN) classifiers perform remarkably well on many problems that require skills which are natural and intuitive to humans. These classifiers have been used in safety-critical systems including autonomous vehicles. For such systems to be trusted it is necessary to demonstrate that the risk factors associated with neural network classification have been appropriately considered and sufficient risk mitigation has been employed. Traditional DNNs fail to explicitly consider risk during their training and verification stages, meaning that unsafe failure modes are permitted and under-reported. To address this limitation, our short paper introduces a work-in-progress approach that (i) allows the risk of misclassification between classes to be quantified, (ii) guides the training of DNN classifiers towards mitigating the risks that require treatment, and (iii) synthesises risk-aware ensembles with the aid of multi-objective genetic algorithms that seek to optimise DNN performance metrics while also mitigating risks. We show the effectiveness of our approach by using it to synthesise risk-aware neural network ensembles for the CIFAR-10 dataset.**

***Index Terms*—deep neural network, risk, risk mitigation**

I. Introduction

Image classification using machine learning has been proposed for use in applications including medical diagnosis and autonomous driving [2], [14]. In particular, deep neural network (DNN) classifiers have shown remarkable results on a range of problems that were previous only thought to be possible using humans. Such problems include speech recognition [4], biomedical imaging and diagnosis [15], and autonomous driving [4]. Despite these advances, image classifiers have limitations that hinder their use in safety-critical applications. These limitations include viewpoint-dependent object variability [5] and overly confident incorrect classification [17].

In safety-critical applications, the risk factors associated with the use of DNNs need to be mitigated appropriately. Nevertheless, current DNN classifier approaches do not explicitly consider risk during the training and verification stages, and make no difference between misclassification of relevant class pairs, where a class pair is formed by the actual and the predicted class. As an example, a bicycle misclassified as a motorbike could lead to an accident if overtaking decisions are made based on the speed of classified vehicle. Traditional approaches treat all misclassifications as equally important, although in many contexts this is not the case.

To addresses this problem, we propose an approach that considers risk during the DNN classifier development process. To that end, we define the risk of misclassification as a function of three factors: (i) the likelihood of encountering each class in the operating environment, (ii) the impact associated with the misclassification of a class as each other class, and (iii) the likelihood of such a misclassification occurring. Our approach enables the calculation of risk ratings for each pair of classes, and therefore the identification of high-risk class pairs—with consideration of the context in which the system is to operate.

Using these risk ratings we train specialist DNN classifiers that mitigate the risks identified, and then synthese ensembles of the risk-aware classifiers by using multi-objective genetic algorithms (GAs). These ensembles seek to optimise DNN performance metrics while also mitigating risk. To demonstrate the effectiveness of our approach, we make use of the commonly used CIFAR-10 dataset [11] and show how a set of ensembles may be constructed which allow a trade-off between risk and traditional DNN performance metrics to be selected.

II. Background

Image classification is the process of allocating a label, from a set of possible labels, to an image. This is a fundamental problem in computer vision, where it forms the basis of localisation, detection, and segmentation [2], [14].

Recently, DNNs have proven to be particularly effective at image classification tasks. A DNN is a layered structure where the output of each layer forms the input to the next. Each layer is constructed from a set of *neurons*, i.e., computational units that calculate a weighted sum of their inputs and apply a non-linear function. The output y_i^l of a unit i in layer l is a function of the outputs of the earlier layer such that $y_i^l = f(\sum_{j=1}^{n_{l-1}} w_j^{l-1} y_j^{l-1} + b_l)$ where n_{l-1} is the number of neurons in layer $l-1$, y_j^{l-1} is output of the j^{th} neuron in layer $l-1$, w_j^{l-1} is a weight associated with the output, b_l is a bias term for layer l, and f is a non-linear activation function.

Initially, the weights are randomly assigned, and then a learning algorithm is used to find the weights which minimise a *loss function* that computes a distance score between the true target and the DNN prediction for a set of data [4].

Consider an n-sample data set $D = \{d_1, d_2, ..., d_n\}$ and a set of classes $C = \{c_1, c_2, ...c_s\}$. Image classification involves assigning a class $c_i \in C$ to each data sample $d_k \in D$. Let $Y = \{y_1, y_2, ..., y_n\}$ be the set of actual classes (ground truth) corresponding to the data set D, where y_k is the actual class of d_k. And let $Y' = \{y'_1, y'_2, ..., y'_n\}$ be the set of predictions made by a DNN model M for each element in D where y'_k is the prediction for the element d_k. The performance of M

$$cm = \left(\begin{array}{cc|ccc} & & \multicolumn{3}{c}{\text{predicted}} \\ & & \text{A} & \text{B} & \text{C} \\ \hline \multirow{3}{*}{\rotatebox{90}{actual}} & \text{A} & 12 & 2 & 1 \\ & \text{B} & 3 & 14 & 6 \\ & \text{C} & 1 & 4 & 9 \end{array} \right)$$

Fig. 1: Sample confusion matrix for a three-class classifier.

can be assessed using a measuring function Φ, which assigns a metric $\phi \in \mathbb{R}$ to the pair (Y, Y'), that is, $(Y, Y') \xrightarrow{\Phi} \phi$ [12].

The performance of a DNN classifier can be evaluated using a *confusion matrix*. Figure 1 shows an example of a confusion matrix for a three-class classifier. The elements on the diagonal of the matrix (highlighted in green) indicate the number of correct predictions made for each class, i.e., the true positive elements (TP), the cells shaded in orange count the number of false negative predictions (FN), and the red cells represent false positive (FP) prediction counts. For instance, the value 2 from row A and column B indicates that 2 data samples with true class A were misclassified as belonging to class B. Traditional DNN performance metrics calculated from the confusion matrix include *recall* and *precision*. Recall measures the effectiveness of a classifier to identify positive labels and is defined as $REC = \frac{TP}{TP+FN}$ [13]. Precision denotes the proportion of predicted positive cases that are correctly real positives, and can be calculated using the following expression $PRC = \frac{TP}{TP+FP}$ [13]. Lastly, the F1 measure is a combination of the above and is defined as the harmonic mean of precision and recall. It can be obtained using $F1 = 2\frac{PRC \cdot REC}{PRC+REC}$ [3]. Derived metrics—such as the *F1 score*—aim to provide a trade-off between precision and recall. Nevertheless, none of these metrics consider the risk associated with modes of misclassification as addressed in our paper.

III. Approach

The steps of our method for taking risk into account in DNN classifiers are depicted in Figure 2 and described as follows.

Step 1: Risk-oblivious training. In this stage, we obtain a set of traditionally trained DNN models. The fact that initially the weights in the DNN are randomly assigned results in different final models, even though the DNN structure remains intact. By generating a set of models we are able to identify those concerns which are common to all models generated from the training set. We term these as *risk-oblivious* because the risk information has not been included during the training of these models and because we use the traditional loss function which gives the same importance to all misclassifications.

Step 2: Risk-aware verification. The objective of this step is to obtain a set of *risk concerns*, which are risk values associated with class pairs deemed to require treatment. We follow the ISO/IEC 31010 risk management standard [8] that provides guidance on how information provided by a domain expert can be included in a risk assessment process by using the following parameters:

1) the likelihood LoE_i of encountering each class i, provided on a five-point ordinal scale—very low (VL), low (L), medium (M), high (H) and very high (VH), and reflecting the likelihood of encountering an instance of a class in a given context;
2) the impact of misclassifying an instance of class i as class j when $i \neq j$, which is defined on the same five-point scale, i.e., $impact_{(i,j)} \in \{VL, L, M, H, VH\}$;
3) the likelihood of misclassification (LoM) thresholds $LoM^0, LoM^{\mathsf{VL}}, LoM^{\mathsf{L}}, LoM^{\mathsf{M}}, LoM^{\mathsf{H}}, LoM^{\mathsf{VH}}$, where $0 = LoM^0 < LoM^{\mathsf{VL}} < LoM^{\mathsf{L}} < LoM^{\mathsf{M}} < LoM^{\mathsf{H}} < LoM^{\mathsf{VH}} < LoM^1 = 1$;
4) the risk threshold τ which specifies the maximum risk level that can be tolerated; the risk associated with misclassifying a data sample of class i as class j needs to be mitigated if its risk level $r_{(i,j)}$ exceeds τ.

TABLE I: Likelihood $OL_{(i,j)}$

LoE_i \ $LoM_{(i,j)}$	VL	L	M	H	VH
VH	L	M	H	VH	VH
H	L	M	M	H	VH
M	L	L	M	M	H
L	VL	L	L	M	M
VL	VL	VL	L	L	L

TABLE II: Risk $r_{(i,j)}$

$OL_{(i,j)}$ \ $impact_{(i,j)}$	VL	L	M	H	VH
VH	L	M	H	VH	VH
H	L	L	M	H	H
M	VL	L	M	M	M
L	VL	L	L	L	L
VL	VL	VL	VL	VL	VL

Given these parameters, and the fraction $p_{(i,j)}$ of data samples of class i from a test data set that are misclassified as class $j \neq i$ by the DNNs from step 1, we first establish the likelihood of misclassification $LoM_{(i,j)}$ for the class pair (i, j) as the unique element from $\{VL, L, M, H, VH\}$ that satisfies $LoM^{pred(LoM_{(i,j)})} < p_{(i,j)} \leq LoM^{LoM_{(i,j)}}$, where the predecessor function $pred$ is defined by $pred(\mathsf{VL}) = 0$, $pred(\mathsf{L}) = \mathsf{VL}$, etc. Next, we compute the overall likelihood $OL_{(i,j)}$ of misclassifying class i as class j by combining LoE_i and $LoM_{(i,j)}$ by using the mapping from Table I. Finally, we use the mapping from Table II to derive the risk level $r_{(i,j)}$ associated with misclassifying class i as class j from $OL_{(i,j)}$ and $impact_{(i,j)}$, and we consider the class pair (i, j) as a *risk concern* if and only if $r_{(i,j)} > \tau$.

As an example, suppose $LoE_i = M$, $LoM_{(i,j)} = VH$ and $impact_{(i,j)} = H$ for a class pair (i, j). Using the mapping from Table I, we obtain $OL_{(i,j)} = H$; then, we combine $OL_{(i,j)}$ and $impact_{(i,j)}$ by using the mapping from Table I, obtaining $r_{(i,j)} = H$. If we further consider $\tau = M$, then all the class pairs with risk level above M, i.e., those corresponding to the highlighted cells from Table II represent the risk concerns. This includes the class (i, j) from our example.

Step 3: Risk-aware training. In this step, the risk concerns previously identified are used to produce sets of risk-aware DNN models that aim to mitigate the risk for the concern they were created for. To achieve that, we modify the cross-entropy loss function by multiplying the result of the classification loss by a weights matrix ω that encodes a penalty value in the class pair whose misclassification risk needs to be mitigated. This amplifies the contribution of the misclassifications to be mitigated to the loss function. As such, the network is "forced" to tweak its weights to reduce the error, and achieves a smaller misclassification value for the concern. The cross-entropy function traditionally assumes a uniform

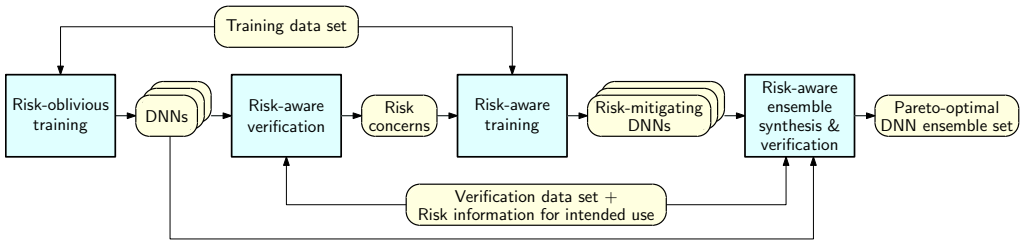

Fig. 2: Four-step method for the synthesis of risk-aware DNN classifiers.

weighting independent of the predicted or actual class. In contrast, our a class-weighted cross entropy is constructed as:

$$\mathcal{L}(\theta) = -\frac{1}{M}\sum_{i=1}^{M}\sum_{n=1}^{N}\omega_n y_n^i \log\left(\hat{p}_n^i\right) \quad (1)$$

where θ represents the DNN parameters to be learnt, M is the number of instances in the data set, y_n^i is the target probability that the i^{th} instance belongs to class n and $\hat{p}_n^i$ is the output of the soft-max for instance i belonging to class n. Note that generally y_n^i is either 1 or 0 depending on whether the instance belongs to the class or not. Finally, ω_n is a weight associated with misclassifying class n. Weighting the class in this way is commonly used to tackle class imbalance in training sets. The resulting *risk-aware models* may be viewed as "experts" in avoiding one form of misclassification. In the next step, we show how these experts are combined with the "generalists" created in the first step of our approach.

Step 4: Risk-aware ensemble synthesis and verification. The final step of our approach combines subsets of the risk-oblivious and the risk-mitigating models obtained in steps 1 and 3, respectively where these subsets are selected based on their performance and risk values. The models in these subsets are fed to a multi-objective genetic algorithm (GA) (we used the DEAP evolutionary framework [6]) whose fitness function seeks to create a DNN ensemble that maximises the F1 score and minimises the $residual\_risk = \sum_{r_{(i,j)}>\tau} res\_risk(LoE_i, impact_{(i,j)}, p_{(i,j)}, \tau)$, subject to a fixed number l of models allowed into the ensemble.[1] The GA is responsible for (i) selecting these models and (ii) providing a set of weights for each of the l models in the ensemble and for each of the s classes: $W_i = \{w_1, w_2, ..., w_s\}$ for $1 \leq i \leq l$. These weights are used to multiply each predicted probability of each model in the ensemble. The output from each model in the ensemble is then weighted before an arg_max function is applied to the sum of predictions for each class. The intuition here is that an *expert* in classifying class i will end up being more highly weighted in the ensemble when predicting class i. The knowledge combination of each of the models in the ensemble is given by $ensemble\_prediction = (Y_1' * W_1 + Y_2' * W_2, ... + Y_i' * W_i)$, where Y_i' is the set of predictions made by a DNN model in the ensemble and W_i the set of weights provided by the GA.

IV. Preliminary Evaluation

To show the effectiveness of our approach, we conducted a series of experiments using the CIFAR-10 dataset [11], which contains $60,000$ colour images belonging to 10 classes including animals (cat, dog, etc.) and vehicles (truck, automobile, etc.). The risk-oblivious and risk-aware models were trained using the CIFAR-10 training images, the models were then combined to train the ensemble and it was tested using the test images [2]. The experimental results are described below and supplementary material is available on our supporting web site https://rb.gy/ytkawe. In step 1 we trained 30 risk-oblivious models with an average F1 score of 0.7902 and an average residual risk of 5.1668. In step 2 we defined our risk threshold $\tau = M$ i.e. we consider risk-concerns all those class pairs with risk level above $\geq H$. We obtained nine risk concerns. In step 3, we trained 30 risk-aware models for each concern, with 10 models built for each $\omega \in \{2, 5, 10\}$ with an average F1 score of 0.7946 and average residual risk of 5.4547.

Finally, in step 4 we synthesised three different risk-aware ensembles varying the number of allowed models l in each of them. We first randomly selected eight models per concern and eight of the risk-oblivious ones from the F1/residual-risk Pareto front for each type of model, obtaining a set of 80 models, these models were given to the GA to synthesise an ensemble allowing two, five and ten models. We did it in this way because, we had the intuition that as the size of the ensemble grows we could obtain classifiers with higher F1 score capable of mitigating the risk-concerns. Figure 3 shows that our intuition is correct; however, we acknowledge that as the number of models grows, so does the time required to synthesise the ensemble (see Table III). Figure 4 shows the Pareto-optimal ensembles found when 10 models are allowed, as expected while generations elapse the F1 score of the classifiers increases and the residual risk is reduced. We also note that this improvement is less and less noticeable, which suggest we approximate the best solution that the GA can produce. We can also notice that all ensemble sizes achieved better F1 score (at the minimum residual risk) than the risk-oblivious models and the risk-aware models: 0.8208, 0.8476, and 0.8549 from the ensembles vs 0.7902 and 0.7946 from the risk-oblivious and the risk-aware models respectively.

[1]The residual risk $res\_risk(LoE_i, impact_{(i,j)}, p_{(i,j)}, \tau)$ for the class pair (i, j) is a positive value that reflects by how much the risk of misclassifying class i as class j exceeds τ if $r_{(i,j)} > \tau$, and zero otherwise. Its calculation is detailed on our supporting web site https://rb.gy/ytkawe.

[2]A full description of how the data is split in the CIFAR-10 dataset can be found at https://www.cs.toronto.edu/~kriz/cifar.html

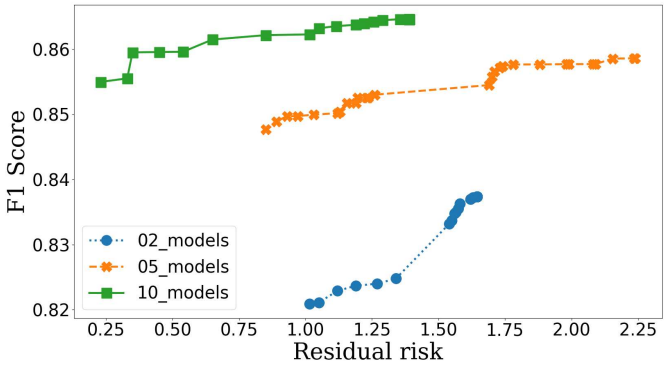

Fig. 3: Pareto-optimal ensembles for different ensemble sizes.

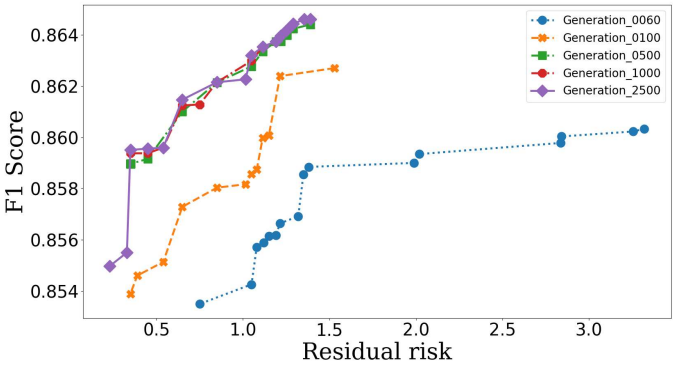

Fig. 4: Ensemble learning history when 10 models are allowed.

TABLE III: Residual risk, F1 score and GA execution time for 2500 generations and ensembles with $l = 2$, 5 and 10 models

	Initial value	Ens. 2	Ens. 5	Ens. 10
Residual risk	4.84	1.015	0.85	0.23
F1 score	0.7902	0.82088	0.8476	0.8549
Time (hours)	-	2.28	4.2	13.18

V. Related work

Various research efforts focus on the importance of weighting for DNNs [1], the problem of unbalanced data along the training process [16], and tackling imbalance classification [10]. These studies mention risk minimisation but refer only to the loss function; a proper risk profile is never used in the training and verification stages. The use of cost matrices to assign a cost to each misclassification based on the distribution of each class is proposed in [16] and [10]. While this is similar to our approach, in the sense that [16] and [10] target particular classes, these solutions focus on modifying the output of a classifier. In contrast, we target the loss function. Additionally, the objective of [16] and [10] is purely to increase the final classification accuracy; the importance of relevant class misclassifications is disregarded in these approaches.

Studies such as [7] use ensemble models to handle concept drift and oversampling, and for dealing with class imbalance; they briefly mention the importance of class misclassification reduction. However, their main focus is not on tackling class misclassification, and their ensembles use simple averaging to combine probability outputs from a set of models. Finally, [9] presents an ensemble for imbalanced classification, and employs an evolutionary algorithm for simultaneous classifier selection and assignment of weights for the fusion process. However, this ensemble synthesis approach does not consider risk; its goal is solely to tackle the imbalance classification. To the best of our knowledge, no existing approaches mitigate the risk associated to different DNN misclassifications explicitly.

VI. Conclusion

We presented an approach for effectively identifying and mitigating risks associated with relevant misclassifications for DNN image classifiers. Its preliminary evaluation indicates that better image classifiers can be constructed by synthesising Pareto-optimal ensembles that include risk-oblivious and risk-mitigation models. Our experiments also show that, as the size of the ensemble increases so does its F1 score and its ability to mitigate risks. Further experiments are needed to identify if at some point the size of the ensemble limits its performance, and whether the GA can find suitable solutions for larger search spaces than in our experiments, and for additional data sets.

References

[1] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *Int. Conf. on ML*, pages 872–881, 2019.

[2] Yashvi Chandola, Jitendra Virmani, HS Bhadauria, and Papendra Kumar. *Deep Learning for Chest Radiographs: Computer-Aided Classification*. Elsevier, 2021.

[3] Davide Chicco and Giuseppe Jurman. The Advantages of the Matthews Correlation Coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.

[4] François Chollet. *Deep Learning with Python*. Manning, 2017.

[5] Dan Claudiu Ciresan et al. Flexible, high performance convolutional neural networks for image classification. In *Twenty-second int. joint conf. on artificial intelligence*, 2011.

[6] Félix-Antoine Fortin et al. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[7] Jing Gao et al. A general framework for mining concept-drifting data streams with skewed distributions. In *Proceedings of the 2007 SIAM int. conf. on data mining*, pages 3–14, 2007.

[8] ISO/IEC Standard 31010:2019. Risk management—risk assessment techniques, 2019.

[9] Bartosz Krawczyk et al. Cost-sensitive decision tree ensembles for effective imbalanced classification. *Applied Soft Computing*, 14:554–562, 2014.

[10] Bartosz Krawczyk and Michał Woźniak. Cost-sensitive neural network with ROC-based moving threshold for imbalanced classification. In *Int. Conf. Intell. Data Eng. and Automated Learning*, pages 45–52, 2015.

[11] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[12] Amalia Luque et al. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, 91:216–231, 2019.

[13] David MW Powers. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[14] Waseem Rawat and Zenghui Wang. Deep Convolutional Neural Networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[15] Guofan Shao et al. Introducing Image Classification Efficacies. *IEEE Access*, 9:134809–134816, 2021.

[16] Shiva Soleymanpour et al. CSCNN: Cost-Sensitive Convolutional Neural Network for encrypted traffic classification. *Neural Processing Letters*, 53(5):3497–3523, 2021.

[17] Bo Yang et al. Lessons learned from accident of autonomous vehicle testing: An edge learning-aided offloading framework. *IEEE Wireless Communications Letters*, 9(8):1182–1186, 2020.