2023

Subnetwork ensembling and data augmentation: Effects on calibration

A. Çağrı Demir

Simon Caton

Pierpaolo Dondio

Funder: Science Foundation Ireland under Grant number 18/CRT/6183.

Received: 20 May 2022 | Revised: 22 December 2022 | Accepted: 6 February 2023

DOI: 10.1111/exsy.13252

ORIGINAL ARTICLE

Expert Systems WILEY

Subnetwork ensembling and data augmentation: Effects on calibration

A. Çağrı Demir[1] | Simon Caton[2] | Pierpaolo Dondio[1]

[1]School of Computer Science, Technological University Dublin, Dublin, Ireland

[2]School of Computer Science, University College Dublin, Dublin, Ireland

Correspondence
A. Çağrı Demir, School of Computer Science, Technological University Dublin, Dublin, Ireland.
Email: cagri.a.demir@mytudublin.ie

Funding information
Science Foundation Ireland, Grant/Award Number: 18/CRT/6183

Abstract

Deep Learning models based on convolutional neural networks are known to be uncalibrated, that is, they are either overconfident or underconfident in their predictions. Safety-critical applications of neural networks, however, require models to be well-calibrated, and there are various methods in the literature to increase model performance and calibration. Subnetwork ensembling is based on the overparametrization of modern neural networks by fitting several subnetworks into a single network to take advantage of ensembling them without additional computational costs. Data augmentation methods have also been shown to enhance model performance in terms of accuracy and calibration. However, ensembling and data augmentation seem orthogonal to each other, and the total effect of combining these two methods is not well-known; the literature in fact is inconsistent. Through an extensive set of empirical experiments, we show that combining subnetwork ensemble methods with data augmentation methods does not degrade model calibration.

KEYWORDS
calibration, data augmentation, ensembles

1 | INTRODUCTION

Deep learning models are starting to be used widely in safety-critical tasks such as autonomous driving (Bojarski et al., 2016) and medical applications. However, to be safely deployed in the real world, these models should output "reliable" predictions, meaning that the distribution of predictions needs to match the empirical distribution of the data. Models which are neither overconfident nor underconfident are called well-calibrated and it is an important characteristic to safely deploy deep learning models (Guo et al., 2017). Besides, real-world data often has different distribution than the data models are trained on. This requires models to be both calibrated and resistant to distributional shifts. Both ensembling and data augmentation techniques have been shown to improve calibration, robustness, and model performance (Havasi et al., 2021; Lakshminarayanan et al., 2017; Shorten & Khoshgoftaar, 2019). However, we still do not fully understand the effects (positive or negative) of combining ensembles with data augmentation methods.

Even a simple averaging of the predictions can help reduce individual model misclassifications and other errors (Fort et al., 2019; Lakshminarayanan et al., 2017). There are different methods for ensembling models which have been shown to be effective in improving accuracy and robustness while not changing the total number of parameters significantly. Among others, subnetwork ensemble frameworks (subnetwork ensemble), BatchEnsemble (Wen et al., 2020) and its variants, and MC dropout (Gal & Ghahramani, 2016) are examples of efficient ensembling methods (Wen et al., 2021). The idea behind training subnetworks comes from sparsity, and the fact that contemporary deep learning models have millions of parameters: are over-parameterized. The overparametrization of deep learning models lead to the lottery ticket hypothesis

(Frankle & Carbin, 2019) and introduction of model pruning methods (Li et al., 2017). Instead of pruning a model to get a subnetwork, subnetwork ensemble models take advantage of available neurons and overparametrization with little structural changes turning a single network into an ensemble of subnetworks. This method enables the generation of ensembles while increasing the total number of parameters by less than 1%. However, training such a model and ensuring independent subnetworks while sharing the main network's parameters with no explicit structural difference is a challenge.

Data augmentation methods encompass a diverse set of methods from basic geometric transformations of images to utilization of GANs (Shorten & Khoshgoftaar, 2019). Prior augmentation methods include cropping, flipping, and rotating, and so forth. Recent augmentation methods such as Cutmix (Yun et al., 2019), MixUp (H. Zhang et al., 2018), and AugMix (Hendrycks et al., 2020) manipulate both the pixels of images and their labels. These augmentations are called Mixed Sample Data Augmentations, and they try to emulate the distribution mismatch between the training and test data by increasing diversity among training images. Increasing the quantity of image datasets with synthetically created images helps to reduce neural networks' errors stemming from overconfidence. Consequently, models using data augmentation are less prone to overfitting and have better generalization capability (Hendrycks et al., 2020). Almost all state-of-art vision models use one or a few data augmentation approaches.

In theory, data augmentation is orthogonal to ensembling (Havasi et al., 2021; Wen et al., 2021). Both ensembling and data augmentation increase accuracy, generalizability, and calibration. However, one cannot directly combine ensembling and data augmentation without further analysis. The findings analysing the interaction between ensembling and data augmentation are mixed in the literature. Wen et al. (2021) shows how combining three ensembling methods (BatchEnsemble, MC dropout, and Deep Ensembles) with two data augmentation methods (Mixup and Augmix) without structural change on the said methods can harm the calibration of the model. Rahaman and Thiery (2021) also show a similar effect while Deep Ensembles used with MixUp augmentation. However, Rame et al. (2021) states that their findings do not confirm the pathology between ensembling and data augmentation, but that combining the two methods increases calibration.

In this paper, we try to clarify this conflict in the literature combining MIMO (Havasi et al., 2021), MixMo (Rame et al., 2021), and Masksembles (Durasov et al., 2021) frameworks (subnetwork ensemble) with data augmentation, and illustrate that this combination does not harm model calibration while increasing accuracy. Moreover, combining ensembles with data augmentation also helps to achieve better uncertainty estimates. We confirmed this behaviour across three different subnetwork ensemble frameworks and two data augmentation methods on three datasets. We also test all models on corrupted Cifar-10 and Cifar-100 datasets (Krizhevsky (2009)) and find consistent results in the presence of corrupted data.

This paper is structured as follows: Section 2 provides some background on the topic, discusses recent approaches in subnetwork ensembles and related augmentation methodologies. Section 3 discusses our experimental approach and gives a brief commentary on key decisions in the experimental workflow. Section 4 presents the main findings and discusses the implications of our findings. Finally, Section 5 concludes the paper and discusses future work as well as how to interpret the results of this paper in future studies.

2 | RELATED WORK

2.1 | Ensemble learning

Ensembling is a technique that takes advantage of diversity among different models to improve their combined performance (Dietterich, 2000). Even simple ensembling (averaging predictions of randomly initialized neural networks) outperforms more complicated models. Lakshminarayanan et al. (2017) show that deep ensembles trained independently improve both accuracy and calibration. Surprisingly, deep ensembles frequently outperform more sophisticated models including Bayesian Neural Networks. Nevertheless, re-training a model with millions of parameters with random initialization can be costly in terms of compute power and time. So, several recent works have attempted to devise efficient ensembling methods with less resources (Fort et al., 2019; Lobacheva et al., 2020; Rahaman & Thiery, 2021). Recently, several ensembling techniques have been proposed in the literature such as BatchEnsemble, MC dropout, and multi-input multi-output frameworks. In this paper, we focus on subnetwork ensemble models.

2.2 | Neural network pruning and subnetwork ensembles

Subnetwork ensemble frameworks are based on the idea of sparsity in neural networks and the fact that contemporary neural network models are overparametrized. One of the drawbacks of over-parametrization is the increased cost of additional memory and computation effort during model training and inference (Hoefler et al., 2021). This enables distilling and pruning methods to create a smaller network from the original network (i.e., a "subnetwork") without sacrificing performance. Subnetwork ensemble models take advantage of these "free" subnetworks in a single neural network and utilize them as an ensemble of networks to improve model performance. Recently several subnetwork ensemble frameworks

have been proposed in the literature. MIMO, MixMo, and Masksembles frameworks are based on this idea: training subnetworks that independently learn the task while utilizing a single model's capacity. The most distinctive feature of subnetwork ensembling is that these models encapsulate diverse subnetworks and train all at once simultaneously. This structure allows them to flexibly exploit the base model's capacity stemming from overparametrization. However, the exact procedure to train models under subnetwork ensemble frameworks and combine the inputs into a shared representation are still active areas of research.

2.3 | Dropout

Dropout was first proposed as a neural network regularization method without an in-depth theoretical grounding (Srivastava et al., 2014). This technique "drops out" neurons in a neural network randomly with a pre-specified probability, hence the name. Although being computationally cheap and intuitively simple, it helps to stabilize training, reduce overfitting and improve generalization performance by removing weights randomly. The original paper proposes that it be used only during training. However, Gal and Ghahramani (2016) associated this method with Bayesian methods and suggested a method they called MC dropout which helps to produce better uncertainty estimates. MC dropout is based on the idea of using dropout at test time and can be viewed as an approximate Bayesian technique. Later, this technique was improved by other studies and several different variations introduced (Srivastava et al., 2014; Gal et al., 2017, Kendall & Gal, 2017; Shen et al., 2021, Z. Zhang et al., 2019).

2.4 | Data augmentation

Data augmentation (DA) increases the training data by introducing small perturbations or transformations (Figure 1); allowing models to be trained on more data. DA helps capture invariant feature transformations and is also used to simulate out-of-distribution data. Therefore, models utilizing DA tend to have better calibration and accuracy resulting in a large uptake of DA in the literature. DA methods also drive the state-of-the-art models for vision tasks (He et al., 2016a). In addition to simple data augmentation methods like random right–left flipping, cropping, and resizing, recent data augmentation methods introduce more complex pixel-wise operations and label manipulations. Cutout (DeVries & Taylor, 2017) uses random occlusion of part of the images whereas CutMix (Yun et al., 2019) replaces a small rectangle area in an image with a rectangle area of another image. Rather than focusing on a specific part of the images, MixUp (H. Zhang et al., 2018) mixes pixels of two images element-wise and combines them. Cubuk et al. (2019) proposes a method based on Reinforcement Learning in which model learns to optimize performance from a group of augmentations. Augmix utilizes a consistency loss to generate a chain of randomly selected augmentations to produce diversity among augmented images (Hendrycks et al., 2020).

2.5 | Summary

The effects of ensembling and data augmentations on image classification tasks are well-studied in the literature. However, we observe limited knowledge and guidance on the total effect when these two seemingly orthogonal methods are combined. Being one of the recent ensembling strategies, subnetwork ensembles achieve ensembling by fitting diverse subnetworks inside a single base network. Recent data augmentation methods also use more complicated processes to generate a diverse set of new images. Accuracy is often the foremost metric targeted by studies. Nevertheless, calibration is also an important metric for model deployment. Hence, in this paper, we seek to provide some clarity on the effects of combining subnetwork ensembles with data augmentation methods and whether this improves model accuracy without harming model calibration.

FIGURE 1 Common data augmentation methods (Rame et al., 2021).

3 | METHODOLOGY

In this paper, we focus on supervised multiclass classification tasks using convolutional neural networks. Our models are based on the ResNet architecture (He et al., 2016a) which uses shortcut connections. We designed our experiments to test model performance and calibration in variety of scenarios. A key part of this paper is that we are not trying to compare models against each other, but rather explore the effects of augmentation on calibration. As such, we do not concern ourselves with whether a specific model is better calibrated than another model or whether there is a distinct advantage (e.g., better accuracy) of using one vs. the other. Instead, we seek to provide guidance on when (and where) augmentation does (or does not) improve model calibration.

3.1 | Experimental design

This paper seeks to understand the impact of combining subnetwork ensemble with data augmentation on calibration. Ensembling and data augmentation are thought to be independent of each other (Havasi et al., 2021; Wen et al., 2021) while both methods are used to enhance model performance. We try to verify Wen et al. (2021)'s hypothesis on ensembling and data augmentation pathology. To do this, we perform a structured $3 \times 3 \times 2$ factorial experimental design consisting of three factors; subnetwork ensemble frameworks (3), data augmentation methods (2), and data sets (3). Although they are not our main focus, we also provide deep ensembles' results acting as a point of comparison. As subnetwork ensemble frameworks, we utilized Multi-input Multi-output (MIMO), MixMo, and Masksembles.

3.1.1 | MIMO

In MIMO (Figure 2), the network takes M inputs and outputs M outputs (predictions) where M is the number of desired subnetworks in a single model. MIMO requires only two changes: the input layer takes M images which are simply stacked images and the output layer has M prediction vectors instead of a single one. In this sense, MIMO uses channel-wise concatenation in pixels for the inputs. These inputs are independently sampled from the training set and require no preprocessing. The base network is trained to predict matching images simultaneously. Each subnetwork learns to disregard features from other images. This ensures the independence of subnetworks. The loss is calculated according to corresponding labels. During testing, the same input is repeated M times, and the outputs are averaged to get the final prediction. Clearly, MIMO does not need the neural network to have large structural changes. In terms of network structure, it is enough to change the first convolutional and last dense layers.

3.1.2 | MixMo

MixMo (Figure 3) has a similar setting to MIMO but instead of channel-wise concatenation of images in pixels, it first encodes each image and then employs a mixing block to combine inputs (Rame et al., 2021). Inspired by mixing data augmentation methods, MixMo uses a generalized multi-input mixing block to combine inputs. Using identity encoding layers and choosing channel-wise concatenation turns the MixMo framework

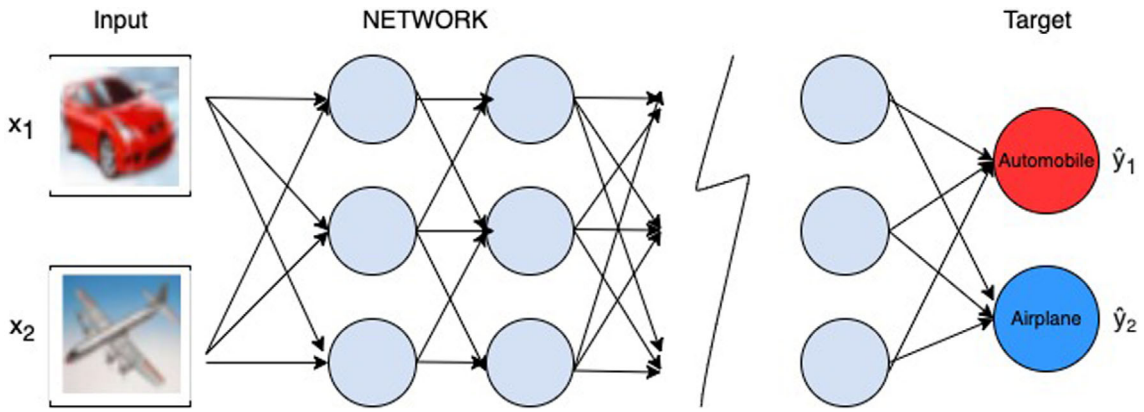

FIGURE 2 MIMO framework with $M = 2$. The network receives two input images, stacks them, and outputs a prediction for each image. All subnetworks share the same base network. At test time, the same input is repeated M times and predictions are averaged to obtain the final prediction.

into MIMO. However, the mixing block is not limited to any specific augmentation method; changing the mixing block results in a different framework. Choosing different augmentations in the mixing block results in different MixMo variants. The two variants presented in the original paper are Cut-MixMo and Linear-MixMo in which CutMix and MixUp (see Figure 1) augmentation methods used to mix input images. Following Rame et al. (2021), we chose Cut-MixMo and refer to it as MixMo from now on, as it is the more performant variant.

Havasi et al. (2021) introduces input repetition and batch repetition during MIMO training. Input repetition helps subnetworks share the same features but degrades diversity among subnetworks. Following the MixMo paper (Rame et al., 2021), we do not utilize input repetition. Batch repetition has a regularization effect on the network training. MIMO finds that the batch repetition value $b = 4$ is optimal, and MixMo also uses $b = 4$. Hence following both papers, we also used batch repetition $b = 4$. One of the core components of subnetwork ensemble frameworks is the number of subnetworks. Since the original network's capacity is limited, as the number of total subnetworks increases, after an optimal number of subnetworks, the performance of the network decreases. Both MIMO and MixMo find that the optimal number of subnetworks is between 2 and 4 for the base models and datasets we utilized. Moreover, the number of subnetworks also increases the training time. We choose the number of subnetworks ($M = 3$) for all models.

3.1.3 | Masksembles

Masksembles (Figure 4) uses parameter masks to introduce a structured way to drop model parameters. The idea behind Masksembles is similar to MC dropout. It basically replaces stochastic sampling in MC dropout and uses fixed number of pre-determined random masks. The diversity

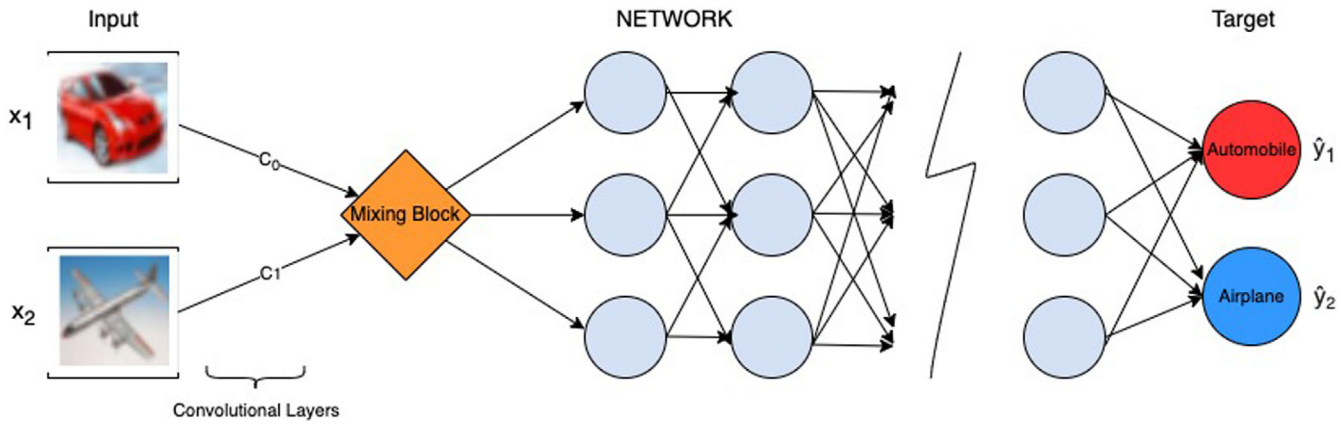

FIGURE 3 MixMo framework with $M = 2$. The network receives two input images, encodes them with convolutional layers, mixes them according to the mixing operation (CutMix or MixUp) and outputs a prediction for each image. All subnetworks share the same base network. At test time, the same input is repeated M times and predictions are averaged to obtain the final prediction.

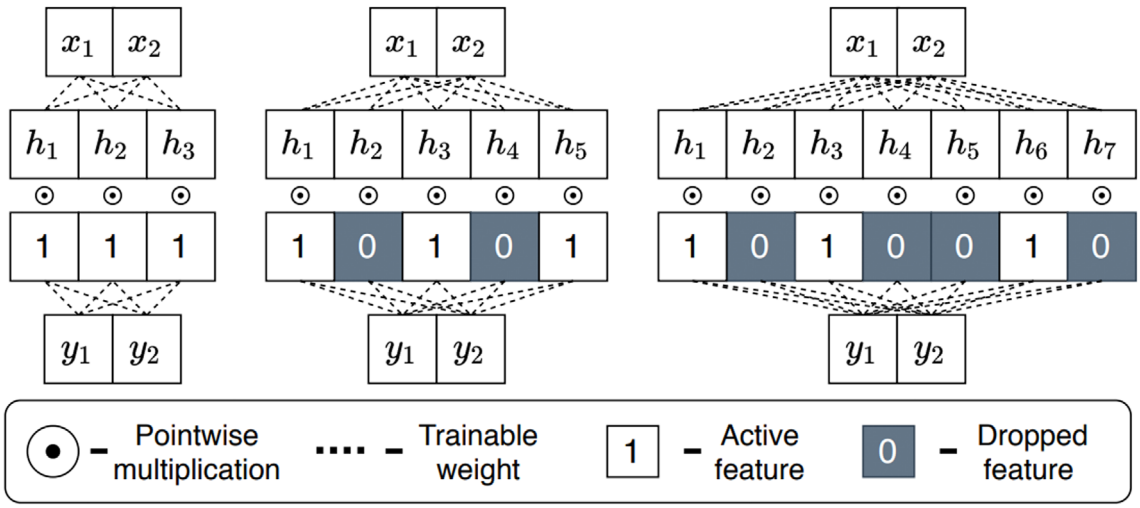

FIGURE 4 A multi-layer perceptron (MLP) example with MaskSembles framework (Durasov et al., 2021). For the above examples number of masks (N) and number of ones in each mask (M) are fixed. The scale parameter (S) is 1.0, 1.7, and 2.3, from left to right respectively.

and the total number of subnetworks are hyperparameters of the framework. There are three key parameters for a Masksemble layer: number of masks (N), number of ones in each mask (M), and scale (S) that controls the amount of overlap given N and M. To match the other subnetwork ensemble framework performance we utilized $N = 4$ and $S = 5$ as suggested by the original paper. Like dropout layers, Masksembles also provides the flexibility of inserting masking layer into the different parts of the network. Following the original paper (Durasov et al., 2021) and dropout paper (Gal & Ghahramani, 2016), Masksemble layers are placed before all the convolutional and fully connected layers.

3.1.4 | Optimization

For all models we used Stochastic Gradient Descent (SGD) with identical hyper-parameters as the corresponding original papers. For Deep Ensembles, we used four models to match the number of subnetworks for MIMO and MixMo. For each model's specification, we trained three randomly initialized models and take the average of metrics. We follow the original papers for learning rate, optimization algorithm, and batch size.

3.1.5 | Base model

Setting aside effects on performance, MIMO, MixMo, and Masksembles frameworks can utilize almost all neural network models as base models, with the ResNet (He et al., 2016a) family being one of the most commonly used. Subnetwork ensemble models need over-parametrization for efficiently fitting diverse subnetworks (Rame et al., 2021). Wide ResNets (Zagoruyko & Komodakis, 2016) are better than the original ResNets to help Subnetwork ensembling exploit model capacity for two reasons (Golubeva et al., 2021; Timpl et al., 2021): (1) a Wide ResNet model has more parameters compared to a ResNet model with the same number of layers, (2) Wide ResNets have more sparsity than a ResNet model with the same number of parameters. Following the original papers (Durasov et al., 2021; Havasi et al., 2021; Rame et al., 2021), all our models are based on a Wide ResNet model for a fair comparison in terms of model performances. For Cifar-10 and Cifar-100, the base model is a WideResNet 28-10 (28 layer ResNet with a widening factor of 10, 36.6 million parameters) (Zagoruyko & Komodakis, 2016) and for TinyImageNet the base model is PreActResNet 18-2 (18 layer ResNet with a widening factor of 2, 44.9 million parameters) (He et al., 2016b).

3.1.6 | Augmentations

To combine with subnetwork ensemble frameworks, we chose two common data augmentation methods: MixUp and CutMix. We go beyond simple data augmentations like flipping, rotation, pixel padding and use recent data augmentation approaches. Indeed, our data augmentation methods fall under the Mixed Sample Data Augmentation, which is the notion of manipulating both images and targets, and creating virtual samples (x_{new}, y_{new}) given two pairs of input images (x_i, y_i) and (x_j, y_j) (see: Section 2). Following original MIMO, MIXMO, and Masksembles papers, data augmentations are performed during training with the probability of 0.5 that a new training sample is generated.

3.1.7 | MixUp

MixUp is a simple data augmentation method which linearly interpolates pixels while manipulating the labels at the same time. The idea behind MixUp is that linear interpolations of feature vectors should lead to linear interpolations of target labels (H. Zhang et al., 2018). By doing so, MixUp extends the training distribution. Given two random samples from training data (x_i, y_i) and (x_j, y_j), when MixUp is applied, we get $(\tilde{x}, \tilde{y})$ by:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j \end{aligned}, \tag{1}$$

where λ is sampled from uniform distribution $\in [0, 1]$.

3.1.8 | CutMix

CutMix creates new images by cutting patches from images and pasting them among training images. CutMix also mixes the true labels proportional to the area of the patches while patching. So a new training sample $(\tilde{x}, \tilde{y})$ is generated by combining two training samples (x_a, y_a) and (x_b, y_b). The combining operations are (Yun et al., 2019):

$$\begin{aligned}\widetilde{x} &= \mathbf{M}\odot x_a + (\mathbf{1}-\mathbf{M})\odot x_b \\ \widetilde{y} &= \lambda y_a + (1-\lambda) y_b\end{aligned}, \tag{2}$$

where M denotes a binary mask indicating where to drop out and fill in from two images, 1 is a binary mask filled with ones, and $\odot$ is element-wise multiplication. Like MixUp, λ is sampled from the uniform distribution $(0,1)$.

3.1.9 | Datasets

We trained all models on the Cifar-10, Cifar-100 (Krizhevsky, 2009), and Tiny ImageNet datasets (Chrabaszcz et al., 2017). Cifar-10 and Cifar-100 datasets both have 60 k images (50 k training and 10 k test images) and 10 and 100 classes respectively. To further push the models we use Tiny ImageNet. Tiny ImageNet (Chrabaszcz et al., 2017) is a downsampled variant of ImageNet as an alternative to the Cifar datasets with 64×64 pixels and with 100 k total images and 200 classes (500 training, 50 validation, and 50 test images per class).

Neural networks encounter a dramatic decrease in their performance when they are tested against out-of-distribution data. After training all models with the matching framework, in addition to IID test sets, we tested all models on corrupted Cifar-10 and Cifar-100 test sets (Hendrycks & Dietterich, 2019). Images in this dataset are perturbed with 19 different common corruption types (e.g., added blur, compression artefacts, frost effects etc.) at five different severity levels. Thus, the Cifar-10 or Cifar-100 test set has $19 \times 5 = 95$ different unseen variations emulating out-of-distribution data. However, a model resilient to a specific type of image corruption at the highest severity level of that corruption (severity level 5) would possibly do so at lower severity levels of the same corruption, that is, severity level 1–4. Likewise, a model with a poor performance against a specific type of image corruption at a low severity level of that corruption would also perform poorly at higher severity levels of that corruption. Therefore, we set the corruption level at 3 for all corruption types across all images to isolate the effect and prevent any over/under-statement of it. A model which improves performance on this should indicate general robustness gain and better calibration (Hendrycks & Dietterich, 2019).

3.2 | Performance metrics: Calibration and uncertainty estimates

Calibration is a notion which measures how a model's predictions match the empirical frequency of the true probabilities (Degroot & Fienberg, 1983). We focus on supervised multi-class classification problems. We say that a model is well calibrated when a prediction of a class with confidence p is correct p% of the time. More formally, we say a model f is calibrated if

$$\forall p \in \Delta : P(Y = y \mid f(X) = p) = p_m, \tag{3}$$

where f is a function mapping every input X to a categorical distribution with the label k, $f(X)$ is a vector in the $(k-1)$-dimensional simplex s.t. $\Delta = \left\{ p \in [0,1]^k \mid \sum_{y=1}^{k} p_y = 1 \right\}$.

A model can have high accuracy yet be a miss-calibrated one. That is calibration and accuracy are two distinct phenomena. Measuring the predictive uncertainty estimates and how well a model is calibrated is a challenging task since the ground truth is not known. Therefore, we utilize two different metrics to measure the calibration: Expected Calibration Error (ECE) and Negative Log-Likelihood (NLL). We also use corrupted Cifar test sets to represent out-of-distribution examples to evaluate model calibration from a domain shift perspective.

By binning the predictions to M equally-spaced intervals and taking a weighted average of each bin's accuracy, Expected Calibration Error (ECE) (Naeini et al., 2015) which measures the absolute difference between accuracy and predictive confidence, is widely used in the literature, and defined as follows:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\mathrm{acc}(B_m) - \mathrm{conf}(B_m)|, \tag{4}$$

where $\mathrm{acc}(B_m)$ is the average probability of the predicted and true class for the bin m and $\mathrm{conf}(B_m)$ is the average confidence within (B_m).

Negative log-likelihood (NLL) is a proper scoring rule (Lakshminarayanan et al., 2017). Scoring rules measure the quality of predictive uncertainty and reward better calibrated predictions (Gneiting & Raftery, 2007). So maximizing likelihood (minimizing NLL) increases calibration. Given a probabilistic model π and n samples, NLL is defined as:

$$\mathscr{L} = -\sum_{i=1}^{n} \log(\pi(y_i|x_i)). \tag{5}$$

4 | EVALUATION

After setting the experimental design and training all models, we tested all models on the respective test sets. We grouped our results for the metrics we track according to datasets. Moreover, we tested all models on corrupted Cifar-10 and Cifar-100. We report the accuracy metrics as well as the ECE and NLL metrics as discussed in Section 3, which are averaged over three independent runs. As our main concern is what happens to subnetwork ensembling models' calibration performance when combined with data augmentation, we will not compare and contrast individual models, but rather discuss how individual models respond to provide more general guidance and comments.

4.1 | Results on Cifar-10/100 and TinyImageNet

Table 1 reports all model results tested on Cifar-10. Subnetwork ensembling frameworks show a performance boost in terms of accuracy compared to the base models. They also improve calibration (lower ECE) and have better uncertainty estimates (lower NLL). When MIMO and MixMo are trained with MixUp and CutMix, model performance across all three metrics also increases. That is, when ensemble models are combined with data augmentation, they better estimate uncertainty (lower NLL) and are better calibrated (lower ECE). We also see a similar trend with Masksembles: metric performance is at least as good, or better.

Table 2 reports results for models trained and tested on Cifar-100. Improvement in the metrics for Cifar-100 is similar to Cifar-10. Combining MixUp or CutMix with one of MIMO, MixMo, and Masksembles makes all models more performant (higher accuracy) and better calibrated (lower ECE & NLL). Combining ensemble models with data augmentation methods results in performance gains across all metrics.

Table 3 reports results for models trained and tested on Tiny ImageNet. We see that results still have the general tendency to be improved when ensembling is combined with data augmentation(s). All three subnetwork ensemble frameworks have higher accuracy and lower calibration error when one of the data augmentations of MixUp and CutMix is added to the training. These results represent strong support of the overall results since Tiny ImageNet dataset has larger images, more labels, and more images than Cifar-10 and Cifar-100.

The test metrics for all three datasets imply that combining subnetwork ensemble frameworks with data augmentation improves accuracy, lowers NLL, and lowers ECE, that is, combining them results in better performance and more calibrated models. However, there is a single exception to this conclusion. When combined with MixMo, MixUp augmentation results in a slight decrease in the calibration (higher NLL & ECE). This situation is true for both Cifar datasets. Nevertheless, for Tiny ImageNet, MixUp behaves in line with the general tendency. This behaviour is consistent across all combinations of subnetwork ensemble and data augmentations. Therefore, combining Subnetwork Ensembles with data augmentations methods does not harm calibration when tested on in-distribution data.

TABLE 1 Performance results for WRN-28-10/CIFAR10.

Model	Data augmentation	Accuracy (↑)	NLL (↓)	ECE (↓)
Base model	–	96.20%	0.149	0.021
Base model	MixUp	96.99%	0.119	0.009
Base model	CutMix	97.47%	0.099	0.012
Deep ensemble	–	96.75%	0.110	0.008
Deep ensemble	MixUp	97.37%	0.100	0.019
Deep ensemble	CutMix	97.93%	0.074	0.006
MIMO	–	96.63%	0.123	0.017
MIMO	MixUp	97.03%	0.114	0.013
MIMO	CutMix	97.56%	0.093	0.011
MixMo	–	97.26%	0.092	0.010
MixMo	MixUp	97.27%	0.106	0.018
MixMo	CutMix	97.42%	0.086	0.010
Masksembles	–	93.25%	0.232	0.027
Masksembles	MixUp	94.31%	0.207	0.027
Masksembles	CutMix	94.73%	0.178	0.017

TABLE 2 Performance results for WRN-28-10/CIFAR100.

Model	Data augmentation	Accuracy (↑)	NLL (↓)	ECE (↓)
Base model	–	81.39%	0.776	0.066
Base model	MixUp	83.30%	0.672	0.013
Base model	CutMix	83.91%	0.658	0.047
Deep ensemble	–	83.42%	0.627	0.027
Deep ensemble	MixUp	85.13%	0.590	0.047
Deep ensemble	CutMix	86.08%	0.536	0.015
MIMO	–	82.75%	0.696	0.064
MIMO	MixUp	83.46%	0.644	0.025
MIMO	CutMix	84.67%	0.562	0.027
MixMo	–	83.86%	0.574	0.038
MixMo	MixUp	84.32%	0.591	0.043
MixMo	CutMix	84.78%	0.540	0.019
Masksembles	–	74.34%	0.983	0.068
Masksembles	MixUp	75.57%	0.902	0.022
Masksembles	CutMix	76.15%	0.869	0.027

TABLE 3 Performance results for PreActResNet-18-2/Tiny ImageNet.

Model	Data augmentation	Accuracy (↑)	NLL (↓)	ECE (↓)
Base model	–	64.75%	2.692	0.123
Base model	MixUp	66.67%	1.789	0.119
Base model	CutMix	67.71%	1.641	0.116
Deep ensemble	–	69.74%	1.508	0.104
Deep ensemble	MixUp	70.19%	1.406	0.083
Deep ensemble	CutMix	70.38%	1.352	0.092
MIMO	–	68.66%	1.691	0.122
MIMO	MixUp	69.02%	1.533	0.096
MIMO	CutMix	70.00%	1.460	0.109
MixMo	–	69.09%	1.405	0.125
MixMo	MixUp	70.81%	1.360	0.078
MixMo	CutMix	70.98%	1.204	0.087
Masksembles	–	56.33%	3.841	0.179
Masksembles	MixUp	58.47%	3.023	0.151
Masksembles	CutMix	60.04%	2.870	0.148

4.2 | Models against image corruptions

Tables 4 and 5 report results when all models are tested against corrupted Cifar datasets. Clearly, compared to IID test sets (uncorrupted), performance of all models across all three metrics degrade. However, still, ensemble models with data augmentations are more calibrated than models without data augmentations.

When compared to the base model, subnetwork ensembles always improve model calibration (lower ECE) with the exception of MIMO (without augmentation). Using an augmentation in addition to subnetwork ensemble almost always improves calibration. The only exception to this is using CutMix with MixMo on Cifar-10, where it also does not improve accuracy. Applying augmentation in addition to Subnetwork Ensembles can boost calibration as much as 3.5× (e.g., MixMo + MixUp). One contrast to the IID dataset is that MixUp helps to enhance both calibration and accuracy more than CutMix when tested against distribution shift. This implies that having an idea (when possible!) of the test data distribution would help to choose which combination to use in model deployment. This is also due to the fact that different Mixed Sample data augmentations can result big train-test distribution gaps (Carratino et al., 2020; Gontijo-Lopes et al., 2021).

TABLE 4 Performance results for WRN-28-10/CIFAR10-corrupted.

Model	Data augmentation	Accuracy (↑)	NLL (↓)	ECE (↓)
Base model	–	78.21%	0.967	0.138
Base model	MixUp	82.97%	0.638	0.062
Base model	CutMix	77.34%	0.973	0.132
Deep ensemble	–	79.72%	0.767	0.083
Deep ensemble	MixUp	84.49%	0.525	0.017
Deep ensemble	CutMix	79.04%	0.766	0.064
MIMO	–	78.03%	0.995	0.146
MIMO	MixUp	84.13%	0.533	0.027
MIMO	CutMix	76.95%	0.972	0.132
MixMo	–	80.55%	0.762	0.094
MixMo	MixUp	84.19%	0.527	0.026
MixMo	CutMix	77.63%	0.981	0.126
Masksembles	–	70.01%	1.210	0.161
Masksembles	MixUp	73.48%	1.090	0.149
Masksembles	CutMix	71.67%	1.100	1.155

TABLE 5 Performance results for WRN-28-10/CIFAR100-corrupted.

Model	Data augmentation	Accuracy (↑)	NLL (↓)	ECE (↓)
Base model	–	53.28%	2.364	0.197
Base model	MixUp	59.03%	1.780	0.074
Base model	CutMix	52.29%	2.388	0.199
Deep ensemble	–	56.01%	2.029	0.111
Deep ensemble	MixUp	61.79%	1.594	0.009
Deep ensemble	CutMix	55.25%	2.065	0.101
MIMO	–	54.21%	2.367	0.208
MIMO	MixUp	56.88%	1.916	0.079
MIMO	CutMix	54.82%	2.090	0.143
MixMo	–	55.98%	2.168	0.172
MixMo	MixUp	58.01%	1.803	0.048
MixMo	CutMix	55.65%	2.007	0.119
Masksembles	–	48.24%	3.560	0.261
Masksembles	MixUp	52.10%	2.440	0.223
Masksembles	CutMix	48.73%	3.310	0.257

To summarize, as in the case for uncorrupted test sets, utilizing data augmentation in addition to subnetwork ensembling help to improve calibration in the case of distribution shift. Out-of-distribution data or distribution shift is relevant for the models to be deployed in real-life situations. Hence, the results showing resilience against image corruption become more relevant for deployment settings and also better highlight overall model performance.

4.3 | Consistent calibration estimator

Calibration metrics do not have a unit or scale, which makes them hard to interpret, especially if there are many models from different frameworks competing with each other. Moreover, the randomness and incurred bias of the calibration metrics may necessitate the use of more stable metrics. Furthermore, as the number of classes in a classification task increases, models scale poorly. Formally, it would be better to have unbiased

TABLE 6 Calibration error estimates (SKCE) for models on Cifar-10 & Cifar-100.

Model	Data augmentation	SKCE-Cifar10 (10^{-4}, ↓)	SKCE-Cifar100 (10^{-4}, ↓)
Base model	–	0.785	0.734
Base model	MixUp	0.209	0.378
Base model	CutMix	0.471	0.501
MIMO	–	0.479	0.649
MIMO	MixUp	0.256	0.333
MIMO	CutMix	0.208	0.222
MixMo	–	0.296	0.310
MixMo	MixUp	0.585	0.271
MixMo	CutMix	0.310	0.229
Masksembles	–	0.892	0.911
Masksembles	MixUp	0.827	0.873
Masksembles	CutMix	0.785	0.812

and consistent estimators of calibration to compare them. To tackle these issues, Widmann et al. (2019) proposes a unifying framework for calibration tests. According to their calibration test, estimators are viewed as calibration test statistics with well-defined bounds. For this reason, we utilize a calibration error called kernel calibration error (KCE), to provide additional insights into model calibration performance. KCE is defined as (Widmann et al., 2019):

$$\mathrm{KCE}[k,g] = \left(E\left[(e_Y - g(X))^T k(g(X),g(X'))(e_{Y'} - g(X'))\right]\right)^{1/2}, \tag{6}$$

if $E[\|k(g(X),g(X))\|] < \infty$ and given k is a matrix-valued kernel as in definition 1 in Widmann et al. (2019), (X',Y') is an independent copy of (X,Y) and e_i denotes the ith unit vector. Based on this let us define a function h s.t.:

$$h_{ij} := \left(e_{Y_i} - g(X_i)^T k(X_i),g(X_j)\right)\left(e_{Y_i} - g(X_j)\right). \tag{7}$$

Hence, the below estimator becomes consistent and unbiased estimator of the squared kernel calibration error $\mathrm{SKCE}[k,g] := \mathrm{KCE}^2[k,g]$:

$$\mathrm{SKCE} = \binom{n}{2}^{-1} \sum_{1 \le i < j \le n} h_{ij}. \tag{8}$$

Table 6 shows SKCE values of each model on Cifar-10 and Cifar-100 test sets. Here we see additional evidence that combining data augmentation and Subnetwork Ensembles does not harm model calibration. In fact, we see a marked difference in the KCE estimator (where lower is better) as defined by Widmann et al. (2019). Again, recall that our intention is to not compare across models, but rather to illustrate that for the models we have experimented with that calibration is improving in the presence of data augmentation methods. Note that at this stage we do not claim that the models are in any way "perfectly" calibrated but that in the search for better calibration, data augmentation approaches certainly seem to help subnetwork ensembles.

5 | CONCLUSION

In this paper, we focused on multi-class classification problems and explore the effect of combining Ensembles with data augmentation on calibration. Our extensive experiments have illustrated that using subnetwork ensemble with data augmentation alone improves model calibration and robustness. More importantly, we find that combining subnetwork ensemble with MixUp or CutMix improves accuracy while not harming model calibration. Thus, adding some clarity to the literature on this point, as we did not observe any trade-off between ensembling and data augmentation for subnetwork ensemble. Rather, in our experiments, we observed that combining subnetwork ensemble and data augmentation improved calibration and uncertainty estimates. Our experiments with benchmark corrupted datasets showed how the findings are also robust with respect

to corruption since the minimum (i.e., best) values for ECE and the NLL were obtained when both data augmentation and subnetwork ensemble were used.

Hence, combining subnetwork ensemble with data augmentation methods for image classification tasks helps to improve performance without sacrificing calibration. This situation signals a divergence on the effects of combining different methods for ensembling with data augmentation. Models trying to boost performance should consider this discrepancy. Exploring this behaviour divergence (as future research) among ensembling methods when combined with data augmentation could yield a better understanding of seemingly uncorrelated methods.

ACKNOWLEDGEMENT

Open access funding provided by IReL.

FUNDING INFORMATION

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 18/CRT/6183. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

ORCID

A. Çağrı Demir https://orcid.org/0000-0002-2759-2267
Simon Caton https://orcid.org/0000-0001-9379-3879
Pierpaolo Dondio https://orcid.org/0000-0001-7874-8762

REFERENCES

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., & Zieba, K. (2016). End to end learning for self-driving cars. *arXiv*. Retrieved 2021-10-13, from http://arxiv.org/abs/1604.07316

Carratino, L., Cissé, M., Jenatton, R., & Vert, J.-P. (2020). On mixup regularization. *arXiv*. https://doi.org/10.48550/ARXIV.2006.06049. Retrieved from https://arxiv.org/abs/2006.06049

Chrabaszcz, P., Loshchilov, I., & Hutter, F. (2017). A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv*. https://doi.org/10.48550/ARXIV.1707.08819. Retrieved from https://arxiv.org/abs/1707.08819

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 113-123). IEEE Computer Society.

Degroot, M. H., & Fienberg, S. E. (1983). The comparison and evaluation of forecasters. *The Statistician*, *32*, 12–22.

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. *arXiv*. https://doi.org/10.48550/ARXIV.1708.04552. Retrieved from https://arxiv.org/abs/1708.04552

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple classifier systems* (pp. 1–15). Springer Berlin Heidelberg.

Durasov, N., Bagautdinov, T., Baque, P., & Fua, P. (2021). Masksembles for uncertainty estimation. In *CVPR*. IEEE/CVF.

Fort, S., Hu, H., & Lakshminarayanan, B. (2019). Deep ensembles: A loss landscape perspective. *arXiv*. https://doi.org/10.48550/ARXIV.1912.02757. Retrieved from https://arxiv.org/abs/1912.02757

Frankle, J., & Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*. OpenReview.net.

Gal, Y., & Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning* (Vol. 48). JMLR.

Gal, Y., Hron, J., & Kendall, A. (2017). Concrete dropout. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 3584–3593). Curran Associates Inc.

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*, 359–378.

Golubeva, A., Gur-Ari, G., & Neyshabur, B. (2021). Are wider nets better given the same number of parameters? In *International Conference on Learning Representations*. OpenReview.net. Retrieved from https://openreview.net/forum?id=_zx8Oka09eF

Gontijo-Lopes, R., Smullin, S., Cubuk, E. D., & Dyer, E. (2021). Tradeoffs in data augmentation: An empirical study. In *International Conference on Learning Representations*. OpenReview.net. Retrieved from https://openreview.net/forum?id=ZcKPWuhG6wy

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 1321–1330). PMLR.

Havasi, M., Jenatton, R., Fort, S., Liu, J. Z., Snoek, J., Lakshminarayanan, B., Dai, A. M., & Tran, D. (2021). Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*. OpenReview.net.

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). IEEE. https://doi.org/10.1109/CVPR.2016.90

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Identity mappings in deep residual networks. In *Computer vision – ECCV* (Vol. 2016). Springer, Cham.

Hendrycks, D., & Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., & Lakshminarayanan, B. (2020). Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations*. OpenReview.net.

Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A. (2021). Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, *22*(241), 1–124. Retrieved from http://jmlr.org/papers/v22/21-0366.html

Kendall, A., & Gal, Y. (2017). What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 5580–5590). Curran Associates Inc.

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images* (Tech. Rep.).

Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017). Pruning filters for efficient convnets. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net.

Lobacheva, E., Chirkova, N., Kodryan, M., & Vetrov, D. P. (2020). *On power laws in deep ensembles*. NeurIPS.

Naeini, M. P., Cooper, G. F., & Hauskrecht, M. (2015). Obtainingwell calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (pp. 2901–2907). AAAI Press.

Rahaman, R., & Thiery, A. H. (2021). Uncertainty quantification and deep ensembles. In A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems*. Curran Associates Inc. Retrieved from https://openreview.net/forum?id=wg_kD_nyAF

Rame, A., Sun, R., & Cord, M. (2021). MixMo: Mixing multiple inputs for multiple outputs via deep subnetworks. In *ICCV*. IEEE.

Shen, Y., Zhang, Z., Sabuncu, M. R., & Sun, L. (2021, January). Real-time uncertainty estimation in computer vision via uncertainty-aware distribution distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* (pp. 707–716). IEEE.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*, 1–48.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958. Retrieved from http://jmlr.org/papers/v15/srivastava14a.html

Timpl, L., Entezari, R., Sedghi, H., Neyshabur, B., & Saukh, O. (2021, July 8). Understanding the effect of sparsity on neural networks' robustness. Presented at the ICML 2021 Workshop on Overparameterization: Pitfalls Opportunities.

Wen, Y., Jerfel, G., Muller, R., Dusenberry, M. W., Snoek, J., Lakshminarayanan, B., & Tran, D. (2021). Combining ensembles and data augmentation can harm your calibration. In *International Conference on Learning Representations*. OpenReview.net.

Wen, Y., Tran, D., & Ba, J. (2020). Batchensemble: An alternative approach to efficient ensemble and lifelong learning. In *International conference on learning representations*. OpenReview.net.

Widmann, D., Lindsten, F., & Zachariah, D. (2019). Calibration tests in multi-class classification: A unifying framework. In *Advances in neural information processing systems* (Vol. 32). Curran Associates Inc.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*. IEEE.

Zagoruyko, S., & Komodakis, N. (2016, September). Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)* (pp. 87.1–87.12). BMVA Press.

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*. OpenReview.net.

Zhang, Z., Dalca, A. V., & Sabuncu, M. R. (2019). Confidence calibration for convolutional neural networks using structured dropout. *arXiv*. Retrieved from https://arxiv.org/abs/1906.09551

AUTHOR BIOGRAPHIES

A. Çağrı Demir received the BSc (Hons.) degree in Mathematics, in 2009, and Master's degree in Financial Engineering, in 2018. He worked as Data Scientist and researcher before starting his PhD at the Technological University Dublin under the Science Foundation Ireland Center for Research Training in Machine Learning ML-LABS program. His research focuses on uncertainty and robustness in deep learning.

Simon Caton received the BSc (Hons.) degree in Computer Science, in 2005, and the PhD degree in Computer Science, in 2010. He worked as Postdoctoral Researcher at the Karlsruhe Institute of Technology, Germany, from 2010 to 2014, and as a Lecturer of Data Analytics at the National College of Ireland between 2014 and 2019. He is currently an Assistant Professor with the School of Computer Science at University College Dublin. His research focuses on the applications of machine learning across the domains of social media, quantum computing, and parallel and distributed computing.

Pierpaolo Dondio received the PhD from Trinity College Dublin and joined the Technological University of Dublin, School of Computer Science as Lecturer in 2012. His main research areas are Multi-Agent Systems, Reasoning Under Uncertainty, and Argumentation Theory. He also investigates Online Communities Behaviour and Dynamics.

How to cite this article: Demir, A. Ç., Caton, S., & Dondio, P. (2023). Subnetwork ensembling and data augmentation: Effects on calibration. *Expert Systems*, *40*(6), e13252. https://doi.org/10.1111/exsy.13252