# Knowledge Distillation by On-the-Fly Native Ensemble

**Xu Lan**[1], **Xiatian Zhu**[2], and **Shaogang Gong**[1]

[1]Queen Mary University of London
[2]Vision Semantics Ltd

## Abstract

Knowledge distillation is effective to train small and generalisable network models for meeting the low-memory and fast running requirements. Existing offline distillation methods rely on a strong pre-trained teacher, which enables favourable knowledge discovery and transfer but requires a complex two-phase training procedure. Online counterparts address this limitation at the price of lacking a high-capacity teacher. In this work, we present an On-the-fly Native Ensemble (ONE) strategy for one-stage online distillation. Specifically, ONE trains only a single multi-branch network while simultaneously establishing a strong teacher on-the-fly to enhance the learning of target network. Extensive evaluations show that ONE improves the generalisation performance a variety of deep neural networks more significantly than alternative methods on four image classification dataset: CIFAR10, CIFAR100, SVHN, and ImageNet, whilst having the computational efficiency advantages.

## 1 Introduction

Deep neural networks have gained impressive success in many computer vision tasks [1; 2; 3; 4; 5; 6; 7]. However, the performance advantages are often gained at the cost of training and deploying resource-intensive networks with large depth and/or width [8; 4; 2]. This leads to the necessary of developing compact yet still discriminative models. Knowledge distillation [9] is one generic meta-solution among the others such as parameter binarisation [10; 11] and filter pruning [12]. The distillation process begins with training a high-capacity *teacher* model (or an ensemble of networks), followed by learning a smaller *student* model which is encouraged to match the teacher's predictions [9] and/or feature representations [13; 14]. Whilst promising the student model quality improvement from aligning with a pre-trained teacher model, this strategy requires a longer training time, significant extra computational cost and large memory (for heavy teacher) with the need for a more complex multi-stage training process, all of which are commercially unattractive [15].

To simplify the distillation training process as above, simultaneous distillation algorithms [16; 15] have been developed to perform online knowledge teaching in a one-phase learning procedure. Instead of pre-training a static teacher model, these methods train simultaneously a set of (typically two) student models which learn from each other in a peer-teaching manner. This approach merges the training processes of the teacher and student models, and uses the peer network to provide the teaching knowledge. Beyond the original understanding of distillation that requires the teacher model larger than the student, they allow to improve any-capacity model performance, leading to a more generically applicable technique. This peer-teaching strategy sometimes even outperforms the teacher based offline distillation, with the plausible reason that the large teacher model tends to overfit the training set and finally provides less information additional to the original training labels [15].

Preprint. Work in progress.

However, existing online distillation has a number of drawbacks: (1) Each peer-student may only provide limited extra information and resulting in suboptimal distillation; (2) Training multiple students significantly increases the computational cost and resource burdens; (3) It requires asynchronous model updating with a notorious need of carefully ordering the operations of prediction and back-propagation across networks. We consider that all the weaknesses are due to the lacking of an appropriate teacher role in the online distillation processing.

In this work, we propose a novel online distillation method that is not only more efficient (lower training cost) and but also more effective (higher model generalisation improvement) as compared to previous alternative methods. In training, the proposed approach constructs a multi-branch variant of a given target network by adding auxiliary branches, creates a native ensemble teacher model from all branches on-the-fly, and learns simultaneously each branch plus the teacher model subject to the same target label constraints. Each branch is trained with two objective loss terms: a conventional softmax cross-entropy loss which matches with the ground-truth label distributions, and a distillation loss which aligns to the teacher's prediction distributions. Comparing with creating a set of student networks, a multi-branch single model is more efficient to train whilst achieving superior generalisation performance and avoiding asynchronous model update. In test, we simply convert the trained multi-branch model back to the original (single-branch) network architecture by removing the auxiliary branches, therefore introducing no test-time cost increase. In doing so, we derive an **On-the-Fly Native Ensemble** (ONE) teacher based simultaneous distillation training approach that not only eliminates the need for pre-training the teacher model in an isolated stage as the offline counterpart and further improves the quality of online distillation.

Experiments on four benchmarks (CIFAR10/100, SVHN, and ImageNet) show that the proposed ONE distillation method enables to train more generalisable target models in a one-phase process than the alternative strategies of offline learning a larger teacher network or simultaneously distilling peer students, the previous state-of-the-art techniques for training small target models.

## 2 Related Work

**Knowledge Distillation** There have been a number of attempts to transfer knowledge between varying-capacity network models [17; 9; 13; 14]. Hinton et al. [9] distilled knowledge from a large pre-trained teacher model to improve the learning of a small target net. The rationale behind distillation is the introduction of extra supervision from teacher model in target model training, beyond a conventional supervised learning objective such as the cross-entropy loss subject to labelled training data. The extra supervision were typically obtained from a pre-trained powerful teacher model in the form of classification probabilities [9], feature representation [13; 14], or inter-layer flow (the inner product of feature maps) [18]. Recently, knowledge distillation has been exploited to distil easy-to-train large networks into harder-to-train small networks [14]. Previously methods based on knowledge distillation is normally offline training, which requires at least two phases of training. The recently proposed deep mutual learning [16] overcomes this limitation by conducting online distillation in one-phase training between two peer student models. Anil et al. [15] further extended this idea to accelerate large scale distributed neural network training. However, existing online distillation methods lacks a strong "teacher" model which limits the efficacy of knowledge discovery and transfer. Like offline counterpart, multiple nets are needed to be trained and therefore computationally expensive. We overcome both limitations by designing a new variant of online distillation training algorithm characterised by simultaneously learning a teacher on-the-fly and the target net and performing batch-wise knowledge transfer in a one-phase procedure.

**Multi-branch Architectures** Multi-branch based neural networks have been widely exploited in computer vision tasks [3; 19; 4]. For example, ResNet [4] can be thought of as a category of two-branch networks where one branch is the identity mapping. Recently, "grouped convolution" [20; 21] has been used as a replacement of standard convolution in constructing multi-branch net architectures. These building blocks are often utilised as templates to build deeper networks for achieving stronger modelling capacity. Whilst sharing the multi-branch principle, our ONE method is fundamentally different from these above existing methods since our objective is to improve the training quality of any given target network, rather than presenting a new multi-branch building block. In other words, our method is a meta network learning algorithm, independent of specific network architectures.

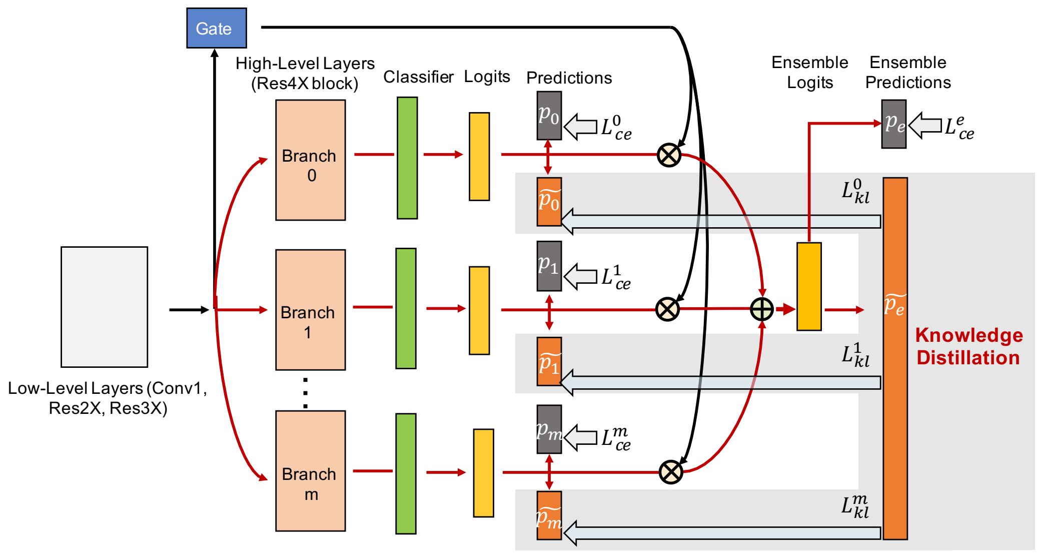


Figure 1: Overview of online distillation training of ResNet-110 by the proposed On-the-fly Native Ensemble (ONE). With ONE, we reconfigure the network by adding $m$ auxiliary branches which share the low-level layers with the target net. Each branch with shared layers makes an individual model, and their ensemble is used to build the teacher model. During a mini-batch training process, we employ the teacher to collect knowledge from individual branch models on-the-fly, which in turn is distilled back to all branches to enhance model learning in a close-loop form. In test, auxiliary branches can be either discarded or kept based on the deployment efficiency requirement.

## 3 Knowledge Distillation by On-the-Fly Native Ensemble

We formulate an online distillation training method based on a concept of On-the-fly Native Ensemble (ONE). For understanding convenience, we take ResNet-110 [4] on CIFAR100 dataset as an example. It is straightforward to apply ONE to other network architectures. For model training, we often have access to $n$ labelled training samples $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_i^n$ with each belonging to one of $C$ classes $y_i \in \mathcal{Y} = \{1, 2, \cdots, C\}$. The network $\boldsymbol{\theta}$ outputs a probabilistic class posterior $p(c|\boldsymbol{x}, \boldsymbol{\theta})$ for a sample $\boldsymbol{x}$ over a class $c$ as:

$$p(c|\boldsymbol{x}, \boldsymbol{\theta}) = f_{sm}(\boldsymbol{z}) = \frac{\exp(\boldsymbol{z}^c)}{\sum_{j=1}^{C} \exp(\boldsymbol{z}^j)}, \quad c \in \mathcal{Y} \tag{1}$$

where $\boldsymbol{z}$ is the logits or unnormalised log probability outputted by the network $\boldsymbol{\theta}$. To train a multi-class classification model, we often adopt the Cross-Entropy (CE) measurement between the predicted and ground-truth label distributions as the objective function:

$$\mathcal{L}_{\text{ce}} = -\sum_{c=1}^{C} \delta_{c,y} \log \Big( p(c|\boldsymbol{x}, \boldsymbol{\theta}) \Big) \tag{2}$$

where $\delta_{c,y}$ is Dirac delta which returns 1 if $c$ is the ground-truth label, and 0 otherwise. With the CE loss, the network is trained to predict the correct class label in the principle of maximum likelihood. To further enhance the model generalisation, we concurrently distil extra knowledge from an on-the-fly native ensemble (ONE) teacher in the training process.

**On-the-Fly Native Ensemble** Overview of the ONE architecture is depicted in Fig 1. The ONE consists of two components: (1) $m$ auxiliary branches with the same configuration (Res4X block and an individual classifier), each of which serves as an independent classification model with shared low-level stages/layers. This is because low-level features are largely shared across different network instances which allows to reduce the training cost. (2) A gate component which learn to ensemble all $(m + 1)$ branches to build a stronger teacher model. It is constructed by one FC layer followed by batch normalisation, ReLU activation, and softmax, and uses the same input features as the branches.

Our ONE method is established based on a multi-branch design specially for model training with several merits: (1) Enable the possibility of creating a strong teacher model without training a set of networks at a high computational cost; (2) Introduce a multi-branch simultaneous learning regularisation which benefits model generalisation (Fig 2(a)); (3) Avoid the tedious need for asynchronous update between multiple networks.

Under the reconfiguration of network, we add a separate CE loss $\mathcal{L}_{\text{ce}}^i$ to all branches which simultaneously learn to predict the same ground-truth class label of a training sample. While sharing the most layers, each can be considered as an independent multi-class classifier given that all of them independently learns high-level semantic representations. Consequently, taking the ensemble of all branches (classifiers) can make a stronger teacher model. One common way of ensembling models is to average individual predictions. This may ignore the diversity and importance variety of member models in the ensemble. We therefore learn to ensemble by the gating component as:

$$\boldsymbol{z}_e = \sum_{i=0}^{m} g_i \cdot \boldsymbol{z}_i \tag{3}$$

where $g_i$ is the importance score of the $i$-th branch's logits $\boldsymbol{z}_i$, and $\boldsymbol{z}_e$ is the logits of the ONE teacher. In particular, we denote the original branch as $i = 0$ for indexing convenience. We train the ONE teacher model with the CE loss $\mathcal{L}_{\text{ce}}^e$ (Eq (2)) same as the branch models.

**Knowledge Distillation** Given the teacher logits for each training sample, we distil knowledge back into all branches in a closed-loop form. For facilitating knowledge transfer, we compute soft probability distributions at a temperature of $T$ for individual branches and the ONE teacher as:

$$\tilde{p}_i(c|\boldsymbol{x}, \boldsymbol{\theta}^i) = \frac{\exp(\boldsymbol{z}_i^c/T)}{\sum_{j=1}^{C}\exp(\boldsymbol{z}_i^j/T)}, \quad \tilde{p}_e(c|\boldsymbol{x}, \boldsymbol{\theta}^e) = \frac{\exp(\boldsymbol{z}_e^c/T)}{\sum_{j=1}^{C}\exp(\boldsymbol{z}_e^j/T)}, \quad c \in \mathcal{Y} \tag{4}$$

where $i$ denotes the branch index, $i = 0, \cdots, m$, $\boldsymbol{\theta}^i$ and $\boldsymbol{\theta}^e$ refer to the parameters of branch and teacher models respectively. Higher values of $T$ lead to more softened distributions.

To quantify the alignment between individual branches and the teacher in their predictions, we use the Kullback Leibler divergence written as:

$$\mathcal{L}_{\text{kl}} = \sum_{i=0}^{m}\sum_{j=1}^{C} \tilde{p}_e(j|\boldsymbol{x}, \boldsymbol{\theta}^e) \log \frac{\tilde{p}_e(j|\boldsymbol{x}, \boldsymbol{\theta}^e)}{\tilde{p}_i(j|\boldsymbol{x}, \boldsymbol{\theta}^i)}. \tag{5}$$

**Overall Loss Function** We obtain the overall loss function for online distillation training by the proposed ONE method as:

$$\mathcal{L} = \sum_{i=0}^{m} \mathcal{L}_{\text{ce}}^i + \mathcal{L}_{\text{ce}}^e + T^2 * \mathcal{L}_{\text{kl}} \tag{6}$$

where $\mathcal{L}_{\text{ce}}^i$ and $\mathcal{L}_{\text{ce}}^e$ are the conventional CE loss terms associated with the $i$-th branch and the ONE teacher, respectively. The gradient magnitudes produced by the soft targets $\tilde{p}$ are scaled by $\frac{1}{T^2}$, so we multiply the distillation loss term by a factor $T^2$ to ensure that the relative contributions of ground-truth and teacher probability distributions remain roughly unchanged. Following [9], we set $T = 3$ in our all experiments.

**Model Training and Deployment** The model optimisation and deployment details are summarised in Alg 1. Unlike the two-phase offline distillation training, the target network and the ONE teacher are trained simultaneously and collaboratively, with the knowledge distillation from the teacher to the target being conducted in each mini-batch and throughout the whole training procedure. Since there is only one multi-branch network rather than multiple networks, we only need to carry out the same stochastic gradient descent through $(m + 1)$ branches, and training the whole network until convergence, as the standard single-model incremental batch-wise training. There is no complexity of asynchronous updating among different networks which is required in deep mutual learning [16].

Once the model is trained, we simply remove all the auxiliary branches and obtain the original network architecture for deployment. Hence, our ONE method does not increase test-time cost. However, if there is less constraint on computation budget and model performance is more important, we can deploy it as an ensemble with all trained branches. We denote this ensemble deployment as **ONE-E**.

**Algorithm 1** Knowledge Distillation by On-the-Fly Native Ensemble

```
 1: Input: Labelled training data $\mathcal{D}$; Training epochs $\tau$; Auxiliary branch number $m$;
 2: Output: Trained target CNN model $\boldsymbol{\theta}^0$, and auxiliary models $\{\boldsymbol{\theta}^i\}_{i=1}^m$;
 3: /* Training */
 4: Initialisation: t=1; Randomly initialise $\{\boldsymbol{\theta}^i\}_{i=0}^m$;
 5: while $t \leq \tau$ do
 6:     Compute predictions of all individual branches $\{p_i\}_{i=0}^m$ (Eq (1));
 7:     Compute the teacher logits (Eq (3));
 8:     Compute the soft targets of all branches and teacher (Eq (4));
 9:     Distil knowledge from the teacher to all branches (Eq (5));
10:     Obtain the final loss function (Eq (6));
11:     Update the model parameters $\{\boldsymbol{\theta}^i\}_{i=0}^m$ by SGD.
12: end
13: /* Testing */
14: Single model deployment: Use $\boldsymbol{\theta}^0$;
15: Ensemble deployment: Use $\{\boldsymbol{\theta}^i\}_{i=0}^m$.
```

## 4 Experiments

**Datasets.** We used four multi-class categorisation benchmark datasets in our evaluations. **(1)** *CIFAR-10* [22]: A natural images dataset that contains 50,000/10,000 training/test samples drawn from 10 object classes (in total 60,000 images). Each class has 6,000 images sized at $32 \times 32$ pixels. **(2)** *CIFAR-100* [22]: A similar dataset as CIFAR10 that also contains 50,000/10,000 training/test images but covering 100 fine-grained classes. Each class has 600 images. **(3)** *SVHN*: The Street View House Numbers (SVHN) dataset consists of 73,257/26,032 standard training/text images and an extra set of 531,131 training images. We used all the training data without data augmentation as [23; 24]. **(4)** *ImageNet*: The 1,000-class ImageNet dataset from ILSVRC 2012 [25] provides 1.2 million images for training, and 50,000 for validation.

**Performance metric.** We adopted the common top-$n$ ($n$=1, 5) classification error rate. For computational cost of model training and test, we used the criterion of floating point operations (FLOPS). For any network trained by ONE, we report the average performance of all branch outputs with standard deviation.

**Experiments setup.** We implemented all networks and model training procedures in Pytorch. For all datasets, we adopted the same experimental settings as [26; 20] for making fair comparisons. We used the SGD with Nesterov momentum and set the momentum to 0.9, following a standard learning rate schedule that drops the rate from 0.1 to 0.01 halfway (50%) through training, and to 0.001 at 75%. For the training budget, CIFAR/SVHN/ImageNet used 300/40/90 epochs respectively. We use three-branch ONE ($m = 2$) design unless stated otherwise.

| Method | CIFAR10 | CIFAR100 | SVHN | Params |
|---|---|---|---|---|
| ResNet-32 [4] | 6.93 | 31.18 | 2.11 | 0.5M |
| ResNet-32 + **ONE** | **5.99**±**0.05** | **26.61**±**0.06** | **1.83**±**0.05** | 0.5M |
| ResNet-110 [4] | 5.56 | 25.33 | 2.00 | 1.7M |
| ResNet-110 + **ONE** | **5.17**±**0.07** | **21.62**±**0.26** | **1.76**±**0.07** | 1.7M |
| ResNeXt-29($8\times64d$) [20] | 3.69 | 17.77 | 1.83 | 34.4M |
| ResNeXt-29($8\times64d$) + **ONE** | **3.45**±**0.04** | **16.07**±**0.08** | **1.70**±**0.03** | 34.4M |
| DenseNet-BC(L=190, k=40) [27] | 3.32 | 17.53 | 1.73 | 25.6M |
| DenseNet-BC(L=190, k=40) + **ONE** | **3.13**±**0.07** | **16.35**±**0.05** | **1.63**±**0.05** | 25.6M |

Table 1: Evaluation of our ONE method on CIFAR and SVHN. Metric: Error rate (%).

### 4.1 Evaluation of On-the-Fly Native Ensemble

**Results on CIFAR and SVHN.** Table 1 compares top-1 error rate performances of four varying-capacity state-of-the-art network models trained by the conventional and our ONE learning algorithms.

| Method | Top-1 | Top-5 |
|---|---|---|
| ResNet-18 [4] | 30.48 | 10.98 |
| ResNet-18 + **ONE** | **29.45±0.23** | **10.41±0.12** |

| Method | Top-1 | Top-5 |
|---|---|---|
| ResNeXt-50 [20] | 22.62 | 6.29 |
| ResNeXt-50 + **ONE** | **21.85±0.07** | **5.90±0.05** |

Table 2: Evaluation of our ONE method on ImageNet. Metric: Error rate (%).

We have these observations: (1) All different networks benefit from the ONE training algorithm, particularly for small models achieving larger performance gains. This suggests the generic superiority of our method for online knowledge distillation from the on-the-fly teacher to the target model. (2) All individual branches have similar performances, indicating that they have made sufficient agreement and exchanged respective knowledge to each other well through the proposed ONE teacher model during training.

**Results on ImageNet.** Table 2 shows the comparative performances on the 1000-classes ImageNet. It is shown that the proposed ONE learning algorithm still yields more effective training and more generalisable models in comparison to vanilla SGD. This indicates that our method can be generically applied to large scale image classification settings.

| Target Network | ResNet-32 | | | ResNet-110 | | |
|---|---|---|---|---|---|---|
| Metric | Error (%) | TrCost | TeCost | Error (%) | TrCost | TeCost |
| KD [9] | **28.83** | 6.43 | 1.38 | N/A | N/A | N/A |
| DML [16] | 29.03±0.22* | **2.76** | 1.38 | **24.10±0.72** | **10.10** | 5.05 |
| **ONE** | **26.61±0.06** | **2.28** | 1.38 | **21.62±0.26** | **8.29** | 5.05 |

Table 3: Comparison with knowledge distillation methods on CIFAR100. "*": Reported results. TrCost/TeCost: Training/test cost, in unit of $10^8$ FLOPs. **Red**/**Blue**: Best and second best results.

## 4.2 Comparison with Distillation Methods

We compared our ONE method with two representative distillation methods: Knowledge Distillation (KD) [9] and Deep Mutual Learning (DML) [16]. For the offline competitor KD, we used a large network ResNet-110 as the teacher and a small network ResNet-32 as the student. For the online methods DML and ONE, we evaluated their performance using either ResNet-32 or ResNet-110 as the target model. We observed from Table 3 that: (1) ONE outperforms both KD (offline) and DML (online) distillation methods in error rates, validating the performance advantages of our method over alternative algorithms when applied to different CNN models. (2) ONE takes the least model training cost and the same test cost as others, and therefore leading to the most cost-effective solution.

| Network | ResNet-32 | | | ResNet-110 | | |
|---|---|---|---|---|---|---|
| Metric | Error (%) | TrCost | TeCost | Error (%) | TrCost | TeCost |
| Snopshot Ensemble [28] | 27.12 | **1.38** | 6.90 | 23.09* | **5.05** | 25.25 |
| 2-Net Ensemble | 26.75 | 2.76 | **2.76** | 22.47 | 10.10 | **10.10** |
| 3-Net Ensemble | **25.14** | 4.14 | 4.14 | **21.25** | 15.15 | 15.15 |
| **ONE-E** | **24.63** | **2.28** | **2.28** | **21.03** | **8.29** | **8.29** |
| **ONE** | 26.61 | 2.28 | 1.38 | 21.62 | 8.29 | 5.05 |

Table 4: Comparison with ensembling methods on CIFAR100. "*": Reported results. TrCost/TeCost: Training/test cost, in unit of $10^8$ FLOPs. **Red**/**Blue**: Best and second best results.

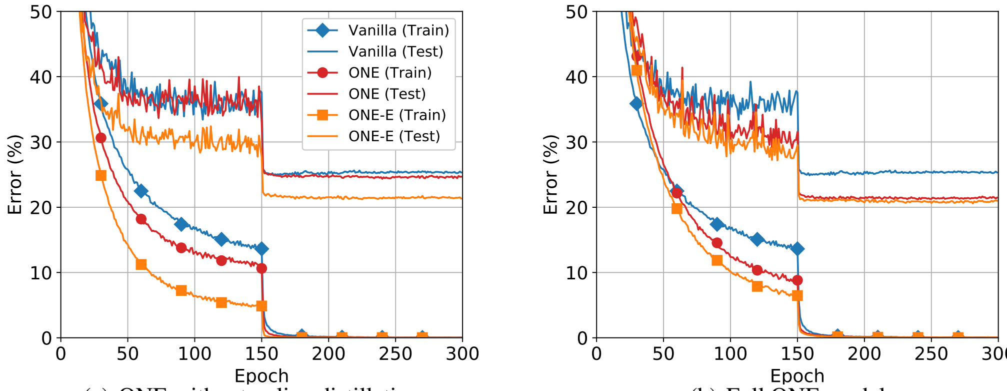
## 4.3 Comparison with Ensembling Methods

Table 4 compares the performances of our multi-branch (3 branches) based model ONE-E and standard ensembling methods. It is shown that ONE-E yields not only the best test error but also allows for most efficient deployment with lowest test cost. These advantages are achieved at second lowest training cost. Whilst Snapshot Ensemble takes the least training cost, its generalisation performance is unsatisfied with a notorious drawback of highest deployment cost.

| Configuration | Full | W/O Online Distillation | W/O Sharing Layers | W/O Gating |
|---|---|---|---|---|
| ONE | **21.62±0.26** | 24.73±0.20 | 22.45±0.52 | 22.26±0.23 |
| ONE-E | 21.03 | 21.84 | **20.57** | 21.79 |

Table 5: Model component analysis on CIFAR100. Network: ResNet-110.

It is worth noting that ONE (without branch ensemble) already outperforms comprehensively 2-Net Ensemble in terms of error rate, training and test cost. Comparing 3-Net Ensemble, ONE is able to approach the generalisation performance whilst having even larger model training and test efficiency advantages.



(a) ONE without online distillation (b) Full ONE model

Figure 2: Effect of online distillation. Network: ResNet-110.

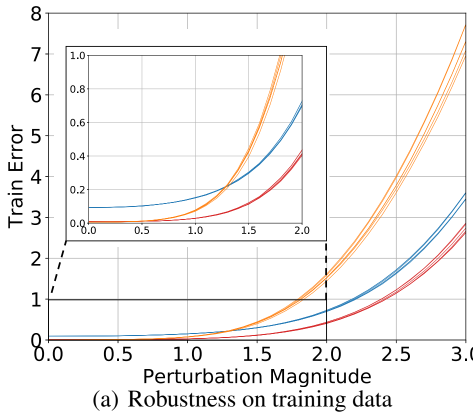
## 4.4 Model Component Analysis

Table 5 shows the benefits of individual ONE components on CIFAR100 using ResNet-110. We have these observations: (1) **Without online distillation** (Eq (5)), the target network suffers a performance drop of $3.11\%$ (24.73-21.62) in test error rate. This performance drop validates the efficacy and quality of ONE teacher in terms of performance superiority over individual branch models. This can be more clearly seen in Fig 2 that ONE teacher fits better to training data and generalises better to test data. Due to the closed-loop design, ONE teacher also mutually benefits from distillation, reducing its error rate from $21.84\%$ to $21.03\%$. With distillation, the target model effectively approaches ONE teacher (Fig 2(a) vs. 2(b)) on both training and test error performance, indicating the success of teacher knowledge transfer. Interestingly, even without distillation, ONE still achieves better generalisation than the vanilla algorithm. This suggests that our multi-branch design brings some positive regularisation effect by concurrently and jointly learning the shared low-level layers subject to more diverse high-level representation knowledge. (2) **Without sharing the low-level layers** not only increases the training cost ($83\%$ increase), but also leads to weaker performance ($0.83\%$ error rate increase). The plausible reason is the lacking of multi-branch regularisation effect as indicated in Fig 2(a). (3) Using average ensemble of branches **without gating** (Eq (3)) causes a performance decrease of $0.64\%$(22.26-21.62). This suggests the benefit of adaptively exploiting the branch diversity in forming the ONE teacher.

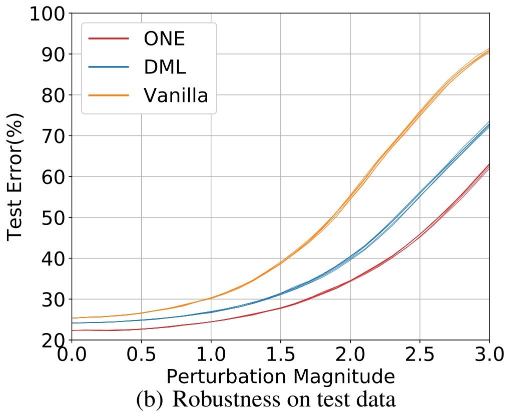
| Branch # | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Error (%) | 31.18 | 27.38 | 26.68 | 26.58 | **26.52** |

Table 6: Benefit of adding branches in ONE on CIFAR100. Network: ResNet-32.

The main experiments using 3 branches in ONE. Table 6 shows that ONE scales well with more branches and the ResNet-32 model generalisation improves on CIFAR100 with the number of branches added during training hence its performance advantage over the independently trained network ($31.18\%$ error rate).



(a) Robustness on training data



(b) Robustness on test data

Figure 3: Robustness test of ResNet-110 solutions found by ONE, DML, and vanilla training algorithms on CIFAR100. Each curve corresponds to a specific perturbation direction $\boldsymbol{v}$.

### 4.5 Model Generalisation Analysis

We aim to give insights on why ONE trained networks yield better generalisation. A few previous studies [29; 30] show that the width of a local optimum is related to model generalisation. The general understanding is that, the surfaces of training and test error largely mirror each other and it is favourable to converge the models to broader optima. As such, the trained model remains approximately optimal even under small perturbations in test time. Next, we exploited this criterion to examine the quality of model solutions $\boldsymbol{\theta}_v$, $\boldsymbol{\theta}_m$, $\boldsymbol{\theta}_o$ discovered by the vanilla, DML and ONE training algorithms respectively. This analysis was conducted on CIFAR100 using ResNet-110.

Specifically, to test the width of local optimum, we added small perturbations to the solutions as $\boldsymbol{\theta}_*(d, \boldsymbol{v}) = \boldsymbol{\theta}_* + d \cdot \boldsymbol{v}, \ * \in \{v, m, o\}$ where $\boldsymbol{v}$ is a uniform distributed direction vector with a unit length, and $d \in [0, 5]$ controls the change magnitude. At each magnitude scale, we further sampled randomly 5 different direction vectors to disturb the solutions. We then tested the robustness of all perturbed models in training and test error rates. The training error is quantified as the cross-entropy measurement between the predicted and ground-truth label distributions. We observed in Fig 3 that: (1) The robustness of each solution against parameter perturbation appears to indicate the width of local optima as: $\boldsymbol{\theta}_v < \boldsymbol{\theta}_m < \boldsymbol{\theta}_o$. That is, ONE seems to find the widest local minimum among three and therefore more likely to generalise better than others. (2) Comparing with DML, vanilla and ONE found deeper local optima with lower training errors. This indicates that DML may probably get stuck in training, therefore scarifying the vanilla's exploring capability for more generalisable solutions to exchange the ability of identifying wider optima. In contrast, our method further improves the capability of identifying wider minima over DML whilst maintaining the original exploration quality. (3) For each solution, the performance changes on training and test data are highly consistent, confirming the earlier observation [29; 30].

## 5 Conclusion

In this work, we presented a novel On-the-fly Native Ensemble (ONE) strategy for improving deep network learning through online knowledge distillation in a one-stage training procedure. With ONE, we can more discriminatively learn both small and large networks with less computational cost, beyond the conventional offline alternatives that are typically formulated to learn better small models alone. Our method is also superior over existing online counterparts due to the unique capability of constructing a high-capacity online teacher to more effectively mine knowledge from the training data and supervise the target network concurrently. Extensive experiments show that a variety of deep networks can all benefit from the ONE approach on four image classification benchmarks. Significantly, smaller networks obtain more performance gains, making our method specially good for low-memory and fast execution scenarios.

## 6 Acknowledgements

This work was partly supported by the China Scholarship Council, Vision Semantics Ltd., the Royal Society Newton Advanced Fellowship Programme (NA150459), and Innovate UK Industrial Challenge Project on Developing and Commercialising Intelligent Video Analytics Solutions for Public Safety (98111-571149).

## References

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

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2015.

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[5] Ross Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision*, 2015.

[6] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[7] Wei Li, Xiatian Zhu, and Shaogang Gong. Person re-identification by deep joint learning of multi-loss classification. In *International Joint Conference of Artificial Intelligence*, 2017.

[8] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv*, 2016.

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv*, 2015.

[10] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542, 2016.

[11] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations*, 2016.

[12] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017.

[13] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662, 2014.

[14] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv*, 2014.

[15] Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. In *International Conference on Learning Representations*, 2018.

[16] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. *arXiv preprint arXiv:1706.00384*, 2017.

[17] Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD*, pages 535–541. ACM, 2006.

[18] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[19] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[20] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.

[21] Gao Huang, Shichen Liu, Laurens van der Maaten, and Kilian Q Weinberger. Condensenet: An efficient densenet using learned group convolutions. *arXiv preprint arXiv:1711.09224*, 2017.

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

[23] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. *arXiv*, 2016.

[24] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial Intelligence and Statistics*, pages 562–570, 2015.

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[26] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, 2016.

[27] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017.

[28] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *International Conference on Learning Representations*, 2017.

[29] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv*, 2016.

[30] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*, 2016.