Hyperparameter Ensembles for Robustness and Uncertainty Quantification

Florian Wenzel, Jasper Snoek, Dustin Tran, Rodolphe Jenatton
fln.wenzel@gmail.com, {jsnoek, trandustin, rjenatton}@google.com

NEURAL INFORMATION PROCESSING SYSTEMS

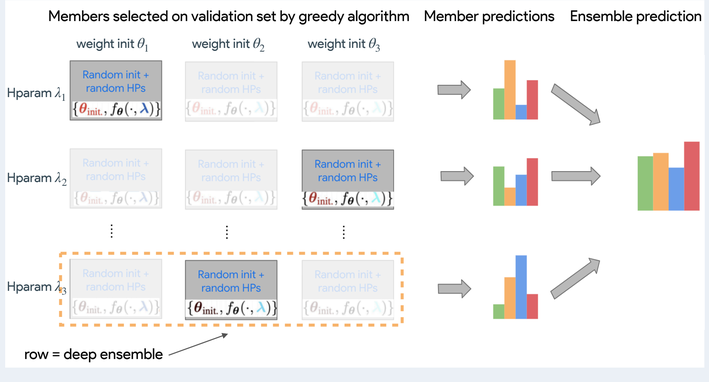
Summary

*We design ensembles of deep networks over hyperparameters and leverage **two sources of diversity:** weight init and different hyperparameters. This improves prediction performance and robustness.*

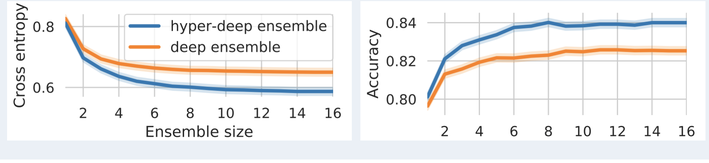
- **Hyper-deep ensembles** extend on deep ensembles and lead to more diverse predictions and SOTA accuracy.
- **Hyper-batch ensembles** is a parameter efficient version that has the same benefits but can be trained as a single model.

Hyper-deep ensembles

- Train κ models by random search (random weight init and random hparam).
- Select M models by greedy ensemble construction algorithm.
- For each selected hparam, retrain for K different weight inits (stratification).
- Run greedy ensemble construction again to select final M models.



Results on CIFAR-100 using Wide-ResNet 28-10

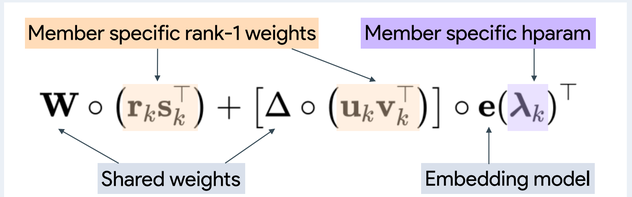


Hyper-Batch Ensembles

How to make it efficient? Hyper-Batch Ensembles capture the behavior of hyper-deep ensembles **within one model.**

Amortized layer parameterization

We amortize the parameterization of a dense layer (conv layers are done similar):



- This combines ideas of batch ensembles (*Wen et al., 2019*), and self-tuning networks (STNs) *(Mackay et al., 2018)*.
- Preserves memory compactness & efficient vectorization.
- Can capture multiple hyperparameters (STNs only covers one hparam).

Ensemble output (avg. of member softmaxes):

$$f_\Theta(\mathbf{x}, \lambda) = \frac{1}{K} \sum_k f_{\Theta_k}(\mathbf{x}, \lambda_k)$$

Training

Model parameters are optimized on the *training set* using the **average member cross entropy** (= the usual loss for single models).

Average member cross entropy — Regularizer

$$\ell\left(\{f_{\Theta_k}(\mathbf{x}, \boldsymbol{\lambda}_k)\}_{k=1}^K, y\right) + \Omega\left(\boldsymbol{\Theta}, \{\boldsymbol{\lambda}_k\}_{k=1}^K\right)$$

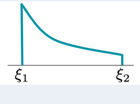
Hyperparameters (more precisely the hyperparameter distribution parameters ξ) are optimized on the *validation set* using the **ensemble cross entropy.** This directly encourages **diversity** between members.

$$\mathbb{E}_{\boldsymbol{\lambda} \sim q_t}\left[\ell_{\text{val}}\left(\{f_\Theta(\mathbf{x}_{\text{val}}, \boldsymbol{\lambda}_k)\}_{k=1}^K, y_{\text{val}}\right) - \tau \mathcal{H}\left[q_t(\lambda)\right]\right]$$

Ensemble cross entropy — Entropy term to prevent collapse of q

Hyperparameter distribution

Log-uniform distribution
$q_t(\lambda_k | \xi_{k,1}, \xi_{k,2})$



Experiments

Prediction performance and calibration on CIFAR-10/100

CIFAR-10: Wide-ResNet 28-10

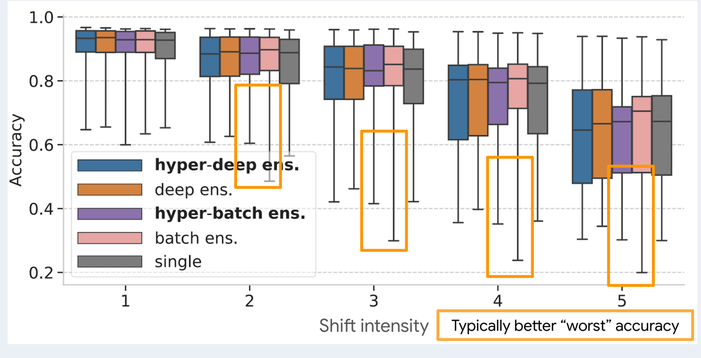
	single (1)	deep ens (4)	hyper-deep ens (4)	batch ens (4)	hyper-batch ens (4)
nll ↓	0.811 ± 0.026	0.678 ± 0.013	**0.636** ± 0.013	0.690 ± 0.005	**0.678** ± 0.005
acc ↑	0.801 ± 0.004	0.819 ± 0.001	**0.831** ± 0.001	**0.819** ± 0.001	**0.820** ± 0.000
ece ↓	0.062 ± 0.001	**0.021** ± 0.002	**0.021** ± 0.002	0.026 ± 0.002	**0.022** ± 0.001
div ↑	–	0.954 ± 0.002	**1.142** ± 0.001	0.761 ± 0.014	**0.996** ± 0.015

CIFAR-100: Wide-ResNet 28-10

	single (1)	deep ens (4)	hyper-deep ens (4)	batch ens (4)	hyper-batch ens (4)
nll ↓	0.152 ± 0.009	**0.108** ± 0.005	**0.108** ± 0.004	0.136 ± 0.001	**0.126** ± 0.001
acc ↑	0.961 ± 0.001	0.968 ± 0.000	**0.969** ± 0.000	**0.963** ± 0.001	**0.963** ± 0.000
ece ↓	0.023 ± 0.005	**0.007** ± 0.003	**0.008** ± 0.002	0.017 ± 0.001	**0.009** ± 0.001
div ↑	–	0.982 ± 0.002	**1.087** ± 0.002	0.444 ± 0.003	**0.874** ± 0.026

Robustness on out-of-distribution data

CIFAR-10 corruptions: each box shows the quartiles summarizing the results across all types of shifts while the error bars give the min/max across different shift types.



Code

Generic layer code (drop-in replacement for your models):
github.com/google/**edward2/**

Experiments:
github.com/google/**uncertainty-baselines/**