HWA: Hyperparameters Weight Averaging Leads To Better Generalization in Bayesian Neural Networks

Belhal Karimi
April 6th 2020

Baidu Research, Cognitive Computing Lab

Agenda

1. SWA: Stochastic Weight Averaging in (Deterministic) Neural Networks
2. Justification of Averaging Benefits
3. HWA: Hyperparameters Weight Averaging in Bayesian Neural Networks

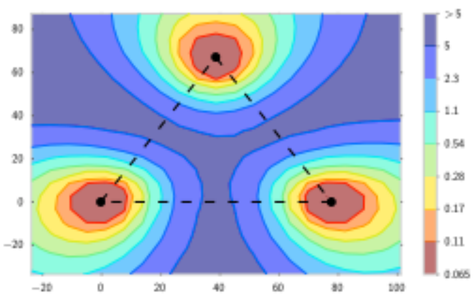
Mode Connectivity

Observations

ResNet-164 on CIFAR-100

- [Garipov et. Al., 2018] shows that three independent trained NNs **(Figure 1)** can be connected via a curve **(Figure 2)**
- Introduces an optimization algorithm to fit such curve
- This curve has the advantage of being of **constant** & **low** loss.
- Thus, there exists a **low loss landscape** between independently trained neural networks.

Figure1



Three optima of three independent trained NN

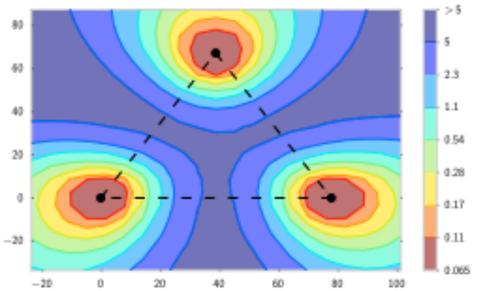
Mode Connectivity

ResNet-164 on CIFAR-100

Observations

- [Garipov et. Al., 2018] shows that three independent trained NNs **(Figure 1)** can be connected via a curve **(Figure 2)**
- Introduces an optimization algorithm to fit such curve
- This curve has the advantage of being of **constant** & **low** loss.
- Thus, there exists a **low loss landscape** between independently trained neural networks.

Figure1



Three optima of three independent trained NN

Connecting Procedure

- Bezier curve provides a convenient parametrization of smooth paths with given endpoints (here two trained neural nets):

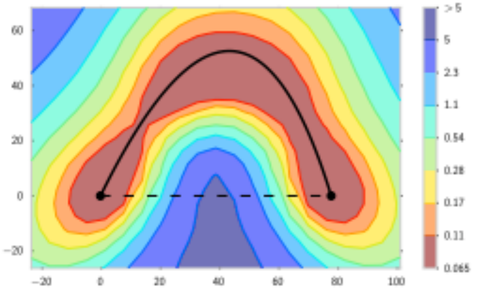
$$\phi_\theta(t) = (1-t)^2\hat{w}_1 + 2t(1-t)\theta + t^2\hat{w}_2, \quad 0 \le t \le 1$$

- Loss to optimize:

$$\hat{\ell}(\theta) = \frac{\int \mathcal{L}\left(\phi_\theta\right) d\phi_\theta}{\int d\phi_\theta} = \frac{\int_0^1 \mathcal{L}\left(\phi_\theta(t)\right) \left\|\phi_\theta'(t)\right\| dt}{\int_0^1 \left\|\phi_\theta'(t)\right\| dt} = \int_0^1 \mathcal{L}\left(\phi_\theta(t)\right) q_\theta(t) dt = \mathbb{E}_{t\sim q_\theta(t)}\left[\mathcal{L}\left(\phi_\theta(t)\right)\right]$$

$\mathcal{L}(w)$ Loss of the NN (cross-entropy)

Figure2

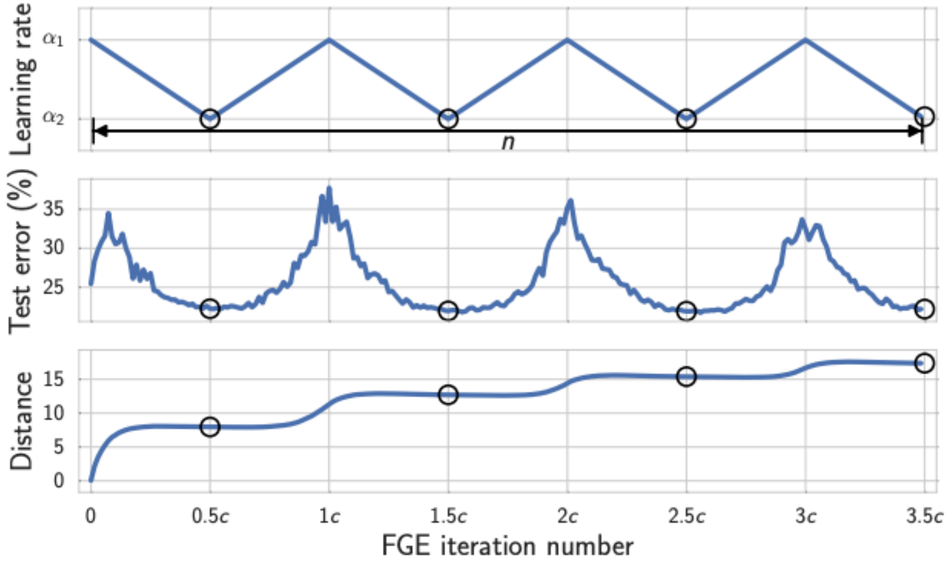


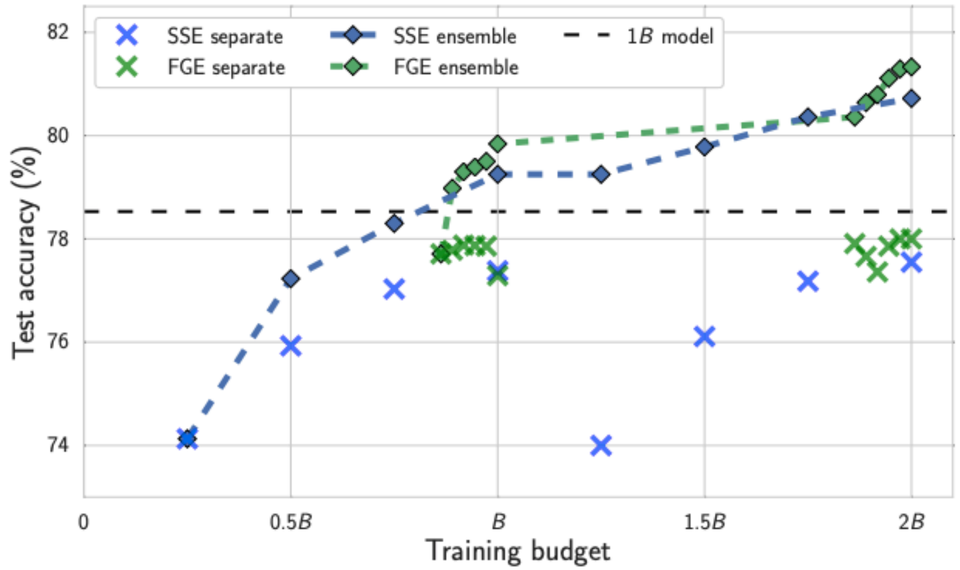
Bezier Curve connecting modes

Ensembling and FGE

Ensembling

- Each mode reaches same training loss **but** can have different testing accuracy
  —> Moving around modes ensures **same** low training loss and can improve accuracy
  —> **Ensembling method to combine all these low loss models**
- [Garipov et. al., 2018] introduces **Fast Geometric Ensembling:**
  1) Start with a trained network (single mode) (no path finding here)
  2) Use a cyclical learning rate, run SGD and takes snapshots of the NN (weights) every time learning rate is the lowest
  3) Then Ensemble of those snapshots





SWA: Stochastic Weight Averaging in NN

Averaging in Optimization

- **Problem with FGE:** Ensembling of K networks (high test time) and not very interpretable
- **Computationally more efficient**: Average in the space weights (and not in the space of trained networks) (Averaging during the estimation VS. averaging of the resulting parameters)
- Even though averaging dates to the seminal paper **[Ruppert, 1988]** it is not used a lot in modern DL.
- Averaging can be seen as an ensembling algorithm along the path of resulting estimates

$$\theta_{n+1} = \theta_n - \gamma_{n+1} g_n(\theta_n) \quad \text{with} \quad \gamma_n = \gamma n^{-\beta}, \beta \in (0,1)$$

$$\text{then} \quad \bar{\theta}_n = \sum_{j=1}^{n} \theta_j$$

SWA: Stochastic Weight Averaging in NN

Averaging in Optimization

- **Problem with FGE:** Ensembling of K networks (high test time) and not very interpretable
- **Computationally more efficient**: Average in the space weights (and not in the space of trained networks) (Averaging during the estimation VS. averaging of the resulting parameters)
- Even though averaging dates to the seminal paper **[Ruppert, 1988]** it is not used a lot in modern DL.
- Averaging can be seen as an ensembling algorithm along the path of resulting estimates

$$\theta_{n+1} = \theta_n - \gamma_{n+1} g_n(\theta_n) \quad \text{with} \quad \gamma_n = \gamma n^{-\beta}, \beta \in (0,1)$$

$$\text{then} \quad \bar{\theta}_n = \sum_{j=1}^{n} \theta_j$$

SWA Algorithm [Izmailov et. al., 2019]

- [Izmailov et. al., 2019] propose equally weighted average of the points traversed by SGD with a cyclical LR
- Stochastic Weight Averaging procedure (SWA):

Algorithm 1 Stochastic Weight Averaging

Require:
weights $\hat{w}$, LR bounds α_1, α_2,
cycle length c (for constant learning rate $c = 1$), number of iterations n

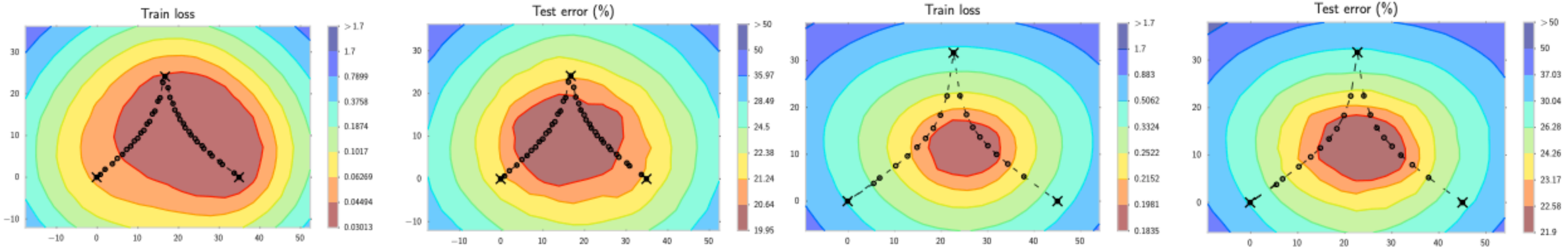
Ensure: w_{SWA}

$w \leftarrow \hat{w}$ {Initialize weights with $\hat{w}$}
$w_{\text{SWA}} \leftarrow w$
for $i \leftarrow 1, 2, \ldots, n$ **do**
  $\alpha \leftarrow \alpha(i)$ {Calculate LR for the iteration}
  $w \leftarrow w - \alpha \nabla \mathcal{L}_i(w)$ {Stochastic gradient update}
  if $\text{mod}(i, c) = 0$ **then**
    $n_{\text{models}} \leftarrow i/c$ {Number of models}
    $w_{\text{SWA}} \leftarrow \frac{w_{\text{SWA}} \cdot n_{\text{models}} + w}{n_{\text{models}} + 1}$ {Update average}
  end if
end for
{Compute BatchNorm statistics for w_{SWA} weights}

SWA: Stochastic Weight Averaging in NN

Results: Training And Generalization

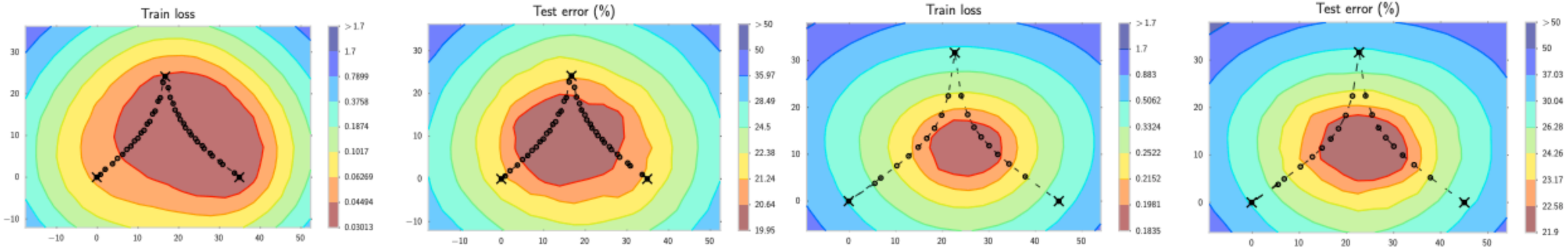
- **Observations:** SGD with constant or cyclical learning *traverses* regions of weight space corresponding to high-performing networks but only moves *around* and never ends in the *optimal* center points
- 1) **Cyclical** (left) is better, 2) **Traverses** but ends around for both and 3) **Shift** train loss from test loss



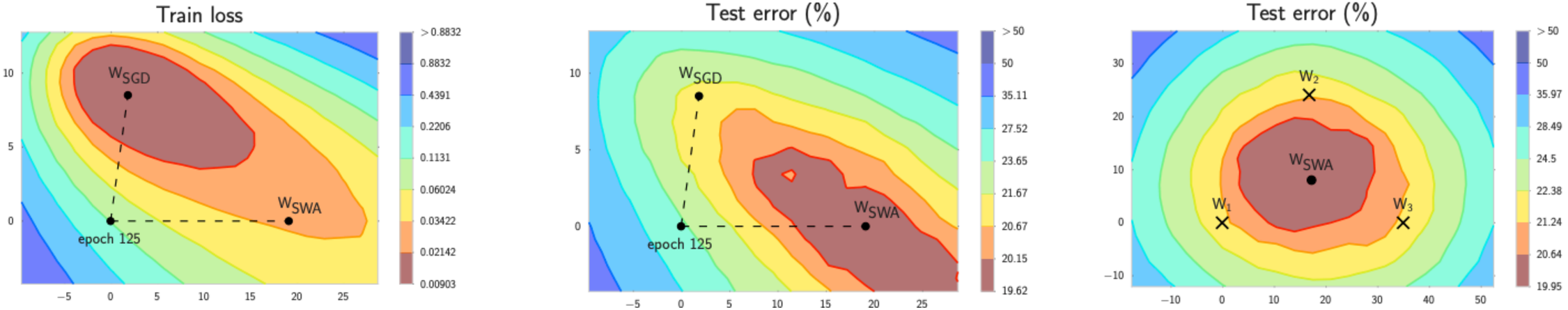
SWA: Stochastic Weight Averaging in NN

Results: Training And Generalization

- **Observations:** SGD with constant or cyclical learning *traverses* regions of weight space corresponding to high-performing networks but only moves *around* and never ends in the *optimal* center points
- 1) **Cyclical** (left) is better, 2) **Traverses** but ends around for both and 3) **Shift** train loss from test loss



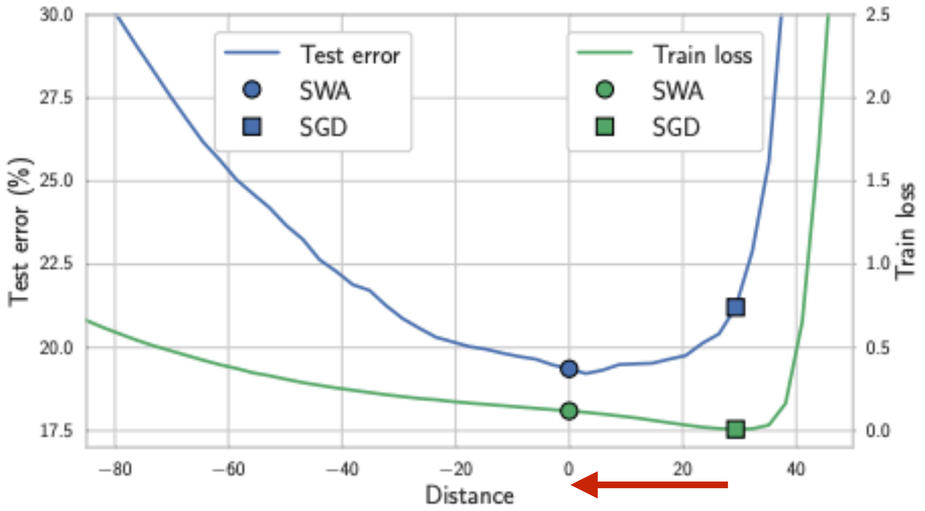
- SWA leads to **worse train error** but better generalization/test error (thanks to the shift of the losses)

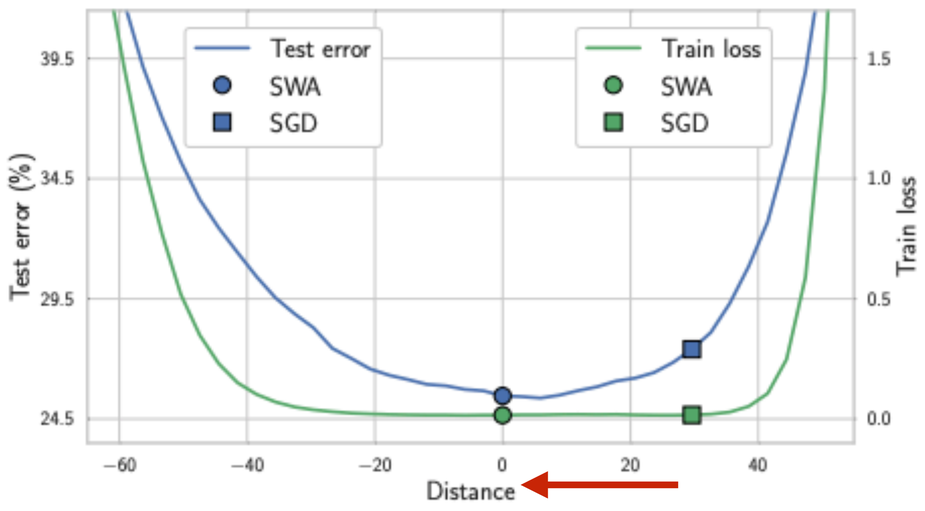


SWA: Stochastic Weight Averaging in NN

Results: Training And Generalization

- **SWA versus SGD:** For ResNet-164 (Left) and VGG-16 (Right) on CIFAR-100

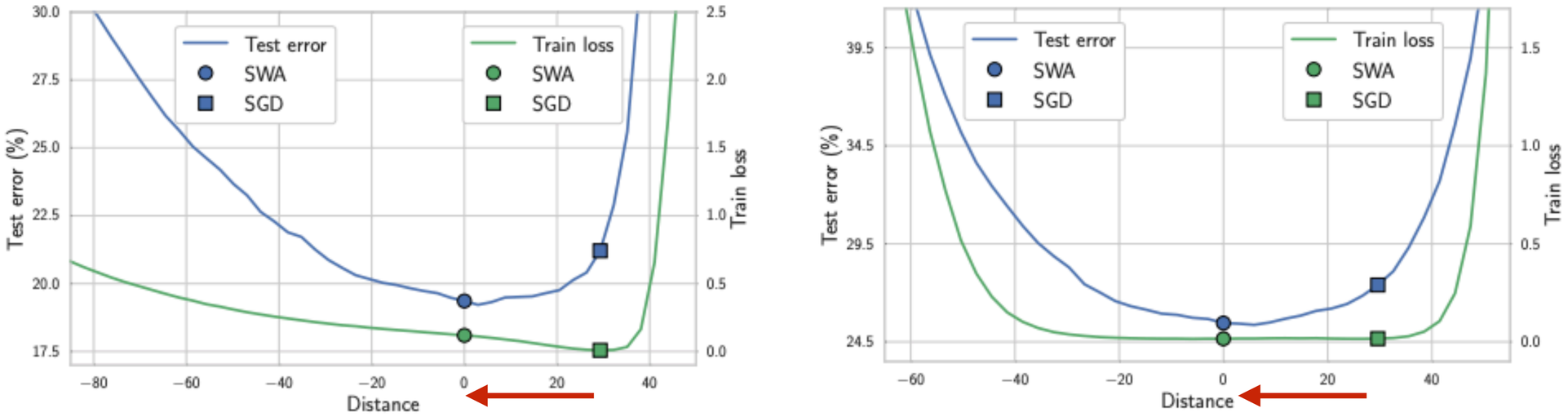




SWA: Stochastic Weight Averaging in NN

Results: Training And Generalization

- **SWA versus SGD:** For ResNet-164 (Left) and VGG-16 (Right) on CIFAR-100



- This brings out the question of **Sharp** and **Flat/Wide** Minima [Keskar et. al., 2017].
- Conjectures that sharp minima leads to worse generalization (test error)
  - See [Bengio, LeRoux et. al., 2019] (Information matrices and generalization) for a connection between generalization and curvature (sharpness) of the loss landscape
- We also see that **introducing a bias** in the SGD solution in the training loss leads to better optima for testing loss (due again to the shift between the two losses)
- **Nonconvex problem too:** averaging well understood for convex [Polyak & Juditsky, 1992]

Justification of Averaging Benefits

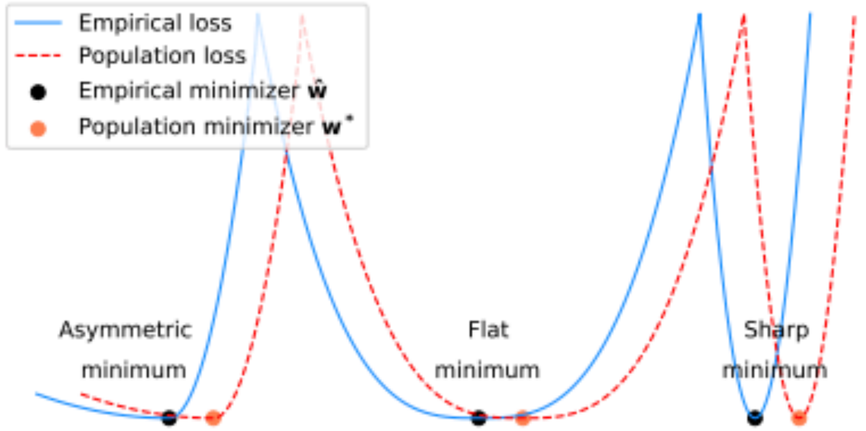
Asymmetric Valleys and Bias Introduction

- « *Asymmetric Valleys: Beyond Sharp and Flat Local Minima* » paper by [He et. al., 2019]
- Focus on **Bias** and considering the **Asymmetry** of the loss landscapes
- 1) Observes that local geometry of the loss function of neural networks is usually asymmetric
- 2) Flatness does affect generalization.
    - In [Keskar et al., 2017] Flat leads to better generalization because Flat means Stable
    - In [He et. al., 2019] Solution introduces a bias that shift to the flat side of the asymmetric valley that leads to better generalization because

Justification of Averaging Benefits

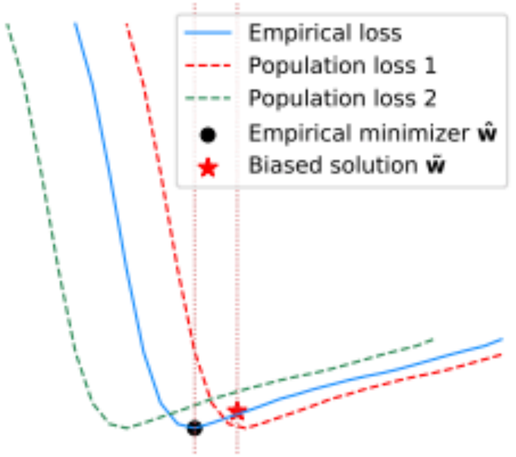
Asymmetric Valleys and Bias Introduction

- « *Asymmetric Valleys: Beyond Sharp and Flat Local Minima* » paper by [He et. al., 2019]
- Focus on **Bias** and considering the **Asymmetry** of the loss landscapes
- 1) Observes that local geometry of the loss function of neural networks is usually asymmetric
- 2) Flatness does affect generalization.
  - In [Keskar et al., 2017] Flat leads to a better generalization because Flat means Stable
  - In [He et. al., 2019] Solution introduces a bias that shift to the flat side of the asymmetric valley that leads to better generalization because of random shift between population and empirical losses



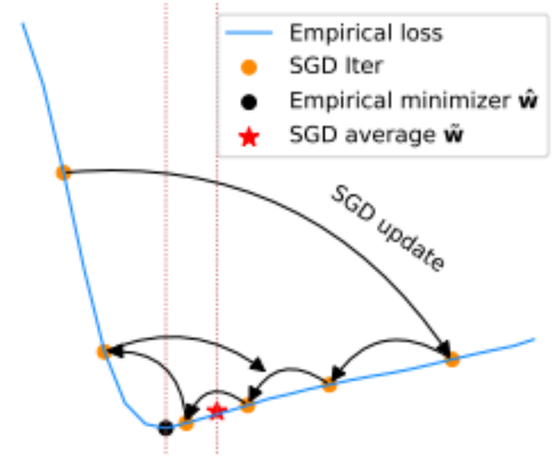
(a)

Three kind of local minima
AV: Some directions loss increases abruptly and slowly along the opposite side



(b)

If there exists a shift, biased solution is better than true training loss minimizer



(c)

SGD spends more time on the flat side —> Average (SWA) should lead on the flat side

Justification of Averaging Benefits

Averaging Leads To ‘Good’ Bias

(H1) Locally Asymmetric Valleys. Where asymmetry is defined by the existence of at least one direction that is asymmetric with respect to the minimizer and the loss

Theorem

Under **(H1)** and considering the SGD updates w_i then the average output noted $\bar{w}$ follows:

$$\mathbb{E}[\bar{w}] > c_0 > 0 \qquad \bar{w} = \frac{1}{n}\sum_{i=1}^{n} w_i$$

Where c_0 depends on the parameters of the asymmetric valleys.

Justification of Averaging Benefits

Averaging Leads To ‘Good’ Bias

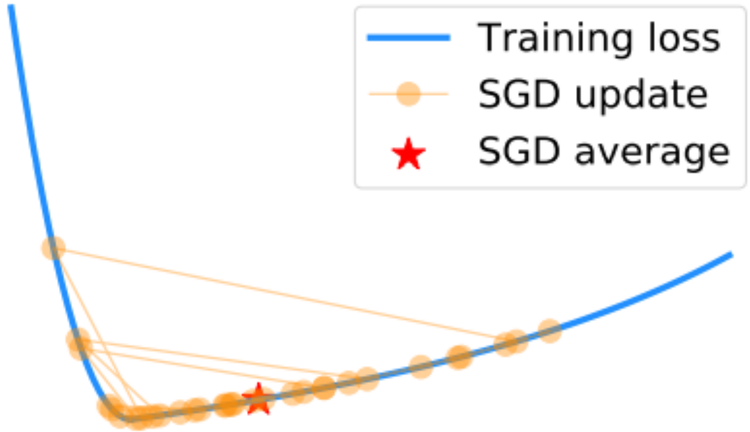
(H1) Locally Asymmetric Valleys. Where asymmetry is defined by the existence of at least one direction that is asymmetric with respect to the minimizer and the loss

Theorem

Under **(H1)** and considering the SGD updates w_i then the average output noted $\bar{w}$ follows:

$$\mathbb{E}[\bar{w}] > c_0 > 0 \qquad \bar{w} = \frac{1}{n}\sum_{i=1}^{n} w_i$$

Where c_0 depends on the parameters of the asymmetric valleys.



Intuitively, gradient/drift term much smaller on the flat side. Thus stays more often there.

- If there is no **oscillation (small learning rate)** and Theorem does not apply, SGD averaging creates more bias on flat sides than sharp sides in expectation. Thus in all the scenarios, taking average of SGD iterates would be beneficial for asymmetric loss function

Justification of Averaging Benefits

Bias Leads To Better Generalization

- **Consider the following assumptions:**

(H1) Locally Asymmetric Valleys.

(H2) Random Shift between Empirical (Train) and Population (Test) Loss.

Theorem

Under **(H1)**, **(H2)** we have:

$$\mathbb{E}_{\boldsymbol{\delta}}\mathbf{L}\left(\hat{\boldsymbol{w}}^*\right) - \mathbb{E}_{\boldsymbol{\delta}}\mathbf{L}\left(\hat{\boldsymbol{w}}^* + \sum_{i=1}^{k} \boldsymbol{l}_i \boldsymbol{u}^i\right) \geq \sum_{i=1}^{k} \left(c_i - 1\right) \boldsymbol{l}_i p_i / 2 - 2k\xi > 0$$

$\hat{\boldsymbol{w}}^*$ is the empirical loss minimizer.

$\{u^i\}_{i=1}^k$ are the k orthogonal directions in $\mathbb{R}^d$ that corresponds to directions of asymmetry

- Theorem states that under the asymmetric case, we should pick a biased solution to minimize the population loss even if the random shift is unknown.

HWA: Hyperparameters Weight Averaging in Bayesian Neural Networks

Settings and Training

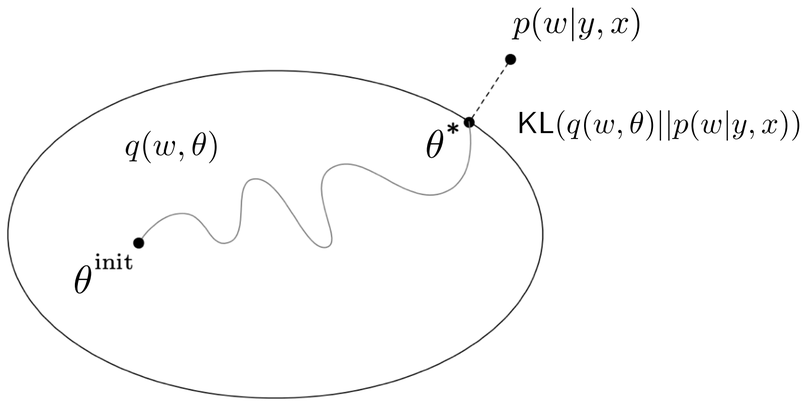
- **Bayesian Neural Networks (BNN):**
- Input-output pairs $((x_i, y_i), 1 \leq i \leq n)$ and w a global latent variable with a prior distribution $\pi(w)$
- Consider the following Classification problem $p(y_i|x_i, w) = \mathrm{Softmax}(f(x_i, w))$ where f is a Neural Network
- Here w are the weights of the Neural Net and are assumed random with distribution $\pi(w) = \mathcal{N}(0, I)$

HWA: Hyperparameters Weight Averaging in Bayesian Neural Networks

Settings and Training

- **Bayesian Neural Networks (BNN):**
- Input-output pairs $((x_i, y_i), 1 \leq i \leq n)$ and w a global latent variable with a prior distribution $\pi(w)$
- Consider the following Classification problem $p(y_i|x_i, w) = \mathrm{Softmax}(f(x_i, w))$ where f is a Neural Network
- Here w are the weights of the Neural Net and are assumed random with distribution $\pi(w) = \mathcal{N}(0, I)$

- **How to train BNNs? —> Variational Inference (VI)**
  - **We want to minimize the KL between the variational candidate $q(w, \theta)$ and the true posterior $p(w|y, x)$**



- KL term is intracable: VI optimizes the Evidence Lower Bound (ELBO)

$$\mathcal{L}(\theta) := -\mathbb{E}_{q(w;\theta)}\big[\log p(y|x, w)\big] + \mathbb{E}_{q(w;\theta)}\big[\log q(w; \theta)/\pi(w)\big]$$

- ELBO is a lower bound of the incomplete log likelihood.
- Maximizing ELBO minimizes the KL

HWA: Hyperparameters Weight Averaging in Bayesian Neural Networks

Current SGD for ELBO Maximization (VI)

$\mathcal{N}(\mu_\ell, \sigma^2 I)$

- SGD on the hyperparameters of the weights (and no longer on the weights themselves). Say candidate is Gaussian

$$\mu_\ell^{k+1} = \mu_\ell^k - \gamma_{k+1} \nabla \mathcal{L}(\mu_\ell^k)$$

Proposed HWA Algorithm

Algorithm 1 HWA: Hyperparameters Weight Averaging

Input: Trained hyperparameters $\hat{\mu}_\ell$ and $\hat{\sigma}$. LR bounds γ_1 and γ_2. Cycle length c.
Initialize the hyperparameters of the weights and $\mu_\ell = \hat{\mu}_\ell$ and $\mu_\ell^{HWA} = \mu_\ell$.
for $k = 0, 1, \ldots$ **do**
  $\gamma \leftarrow \gamma(k)$ (Cyclical LR for the iteration)
  $\mu_\ell^{k+1} \leftarrow \mu_\ell^k - \gamma \nabla \mathcal{L}(\mu_\ell^k)$
  if $\mathrm{mod}(k, c) = 0$ **then**
    $n_{\text{models}} \leftarrow k/c$
    $\mu_\ell^{HWA} \leftarrow \frac{n_{\text{models}} \mu_\ell^{HWA} + \mu_\ell^{k+1}}{n_{\text{models}} + 1}$
  end if
end for

Perspectives

Investigation and To-Do

- **Derive New Algorithm**

- **Run some BNNs (Bayesian LeNet-5 and Bayesian ResNet-18) on MNIST and CIFAR**
  - Observe Mode Connectivity?
  - Observe Bias?
  - Observe Better Generalization?

- **Statistical Guarantees to understand if averaging hyperparameters have same impact on the Generalization error**

Thank You!