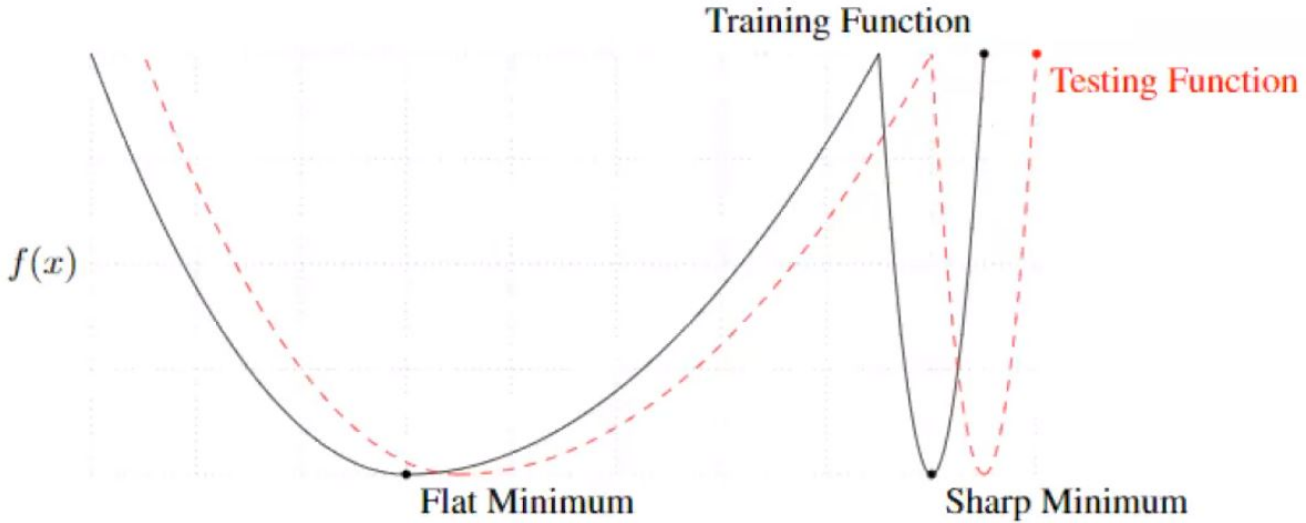
Широкие и узкие оптимумы

Методы получения более широких оптимумов (SAM и SWA)

План

1. Разобраться в широких и узких оптимумы, определить ширину оптимума, понять почему широкие оптимумы лучше
2. Разобраться в методах SAM и SWA

Широкие и узкие оптимумы



по оси Y отмечаются значения функции потерь, а по оси X значения параметров

Проблема: как определить ширину оптимума в многомерном случае

Определение ширины оптимума

1. Объем области, в которой значение функции потерь не сильно отличаются друг от друга (Hochreiter & Schmidhuber, 1997)
2. Максимальное значение функции потери в окрестности найденного оптимума (Кескар и др., 2017)
3. Собственное значение Гессиана в точках минимума (Яо и др., 2018)

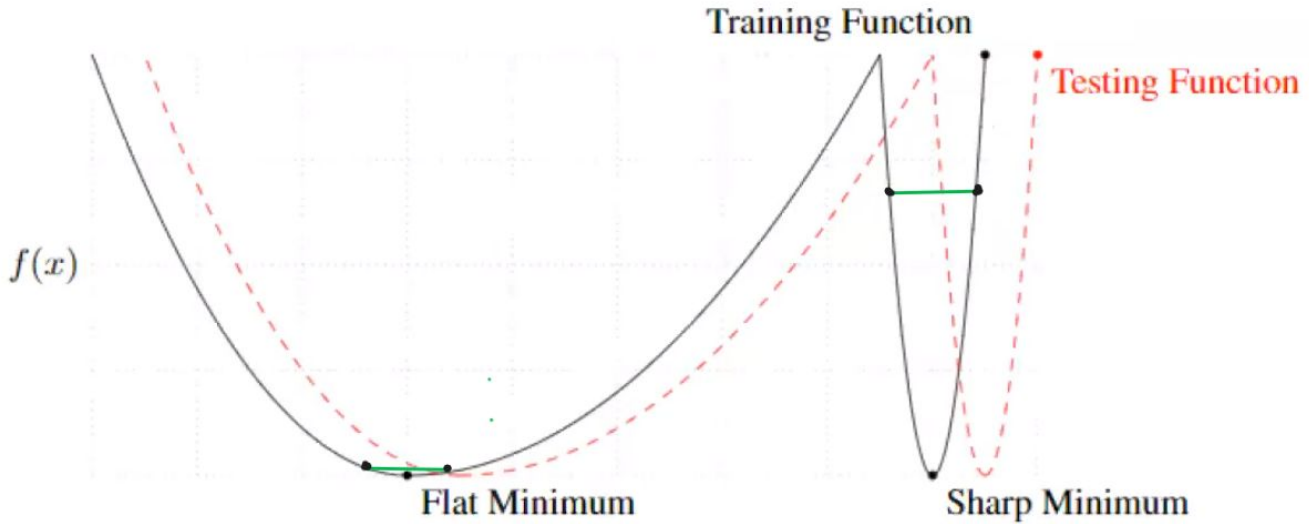
Очевидное наблюдение

Чем оптимум шире — тем выше обобщая способность и адаптивность модели

Глобальная идея – искать широкие минимумы

SAM (Sharpness-Aware Minimization)

1. Метод для эффективного повышения обобщенности модели
2. Основная идея – минимизировать функцию потерь в некоторой окрестности



Вводим новую функцию потерь с гиперпараметром $p \in [1, \infty]$

$$L_{\mathcal{S}}^{SAM}(\boldsymbol{w}) \triangleq \max_{||\boldsymbol{\epsilon}||_p \leq \rho} L_S(\boldsymbol{w} + \boldsymbol{\epsilon}),$$

Задача, лежащая в основе метода SAM:

$$\min_{\boldsymbol{w}} L_{\mathcal{S}}^{SAM}(\boldsymbol{w}) + \lambda ||\boldsymbol{w}||_2^2$$

Эффективный способ посчитать градиент

$$\nabla_{\boldsymbol{w}} L_{\mathcal{S}}^{SAM}(\boldsymbol{w}) \approx \nabla_{\boldsymbol{w}} L_{\mathcal{S}}(w)|_{\boldsymbol{w}+\hat{\boldsymbol{\epsilon}}(\boldsymbol{w})}.$$

где

$$\hat{\boldsymbol{\epsilon}}(\boldsymbol{w}) = \rho \, \text{sign}\left(\nabla_{\boldsymbol{w}} L_{\mathcal{S}}(\boldsymbol{w})\right) |\nabla_{\boldsymbol{w}} L_{\mathcal{S}}(\boldsymbol{w})|^{q-1} \Big/ \left(\|\nabla_{\boldsymbol{w}} L_{\mathcal{S}}(\boldsymbol{w})\|_q^q \right)^{1/p}$$

$$\boldsymbol{\epsilon}^*(\boldsymbol{w}) \triangleq \underset{\|\boldsymbol{\epsilon}\|_p \leq \rho}{\arg\max} L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon}) \approx \underset{\|\boldsymbol{\epsilon}\|_p \leq \rho}{\arg\max} L_{\mathcal{S}}(\boldsymbol{w}) + \boldsymbol{\epsilon}^T \nabla_{\boldsymbol{w}} L_{\mathcal{S}}(\boldsymbol{w}) = \underset{\|\boldsymbol{\epsilon}\|_p \leq \rho}{\arg\max} \boldsymbol{\epsilon}^T \nabla_{\boldsymbol{w}} L_{\mathcal{S}}(\boldsymbol{w}).$$

Реализация метода SAM

Input: Training set $\mathcal{S} \triangleq \cup_{i=1}^{n}\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$, Loss function $l : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$, Batch size b, Step size $\eta > 0$, Neighborhood size $\rho > 0$.
Output: Model trained with SAM
Initialize weights $\boldsymbol{w}_0$, $t = 0$;
while *not converged* **do**
  Sample batch $\mathcal{B} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), ...(\boldsymbol{x}_b, \boldsymbol{y}_b)\}$;
  Compute gradient $\nabla_{\boldsymbol{w}} L_{\mathcal{B}}(\boldsymbol{w})$ of the batch's training loss;
  Compute $\hat{\boldsymbol{\epsilon}}(\boldsymbol{w})$ per equation 2;
  Compute gradient approximation for the SAM objective (equation 3): $\boldsymbol{g} = \nabla_w L_{\mathcal{B}}(\boldsymbol{w})|_{\boldsymbol{w}+\hat{\boldsymbol{\epsilon}}(\boldsymbol{w})}$;
  Update weights: $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{g}$;
  $t = t + 1$;
end
return $\boldsymbol{w}_t$

Algorithm 1: SAM algorithm

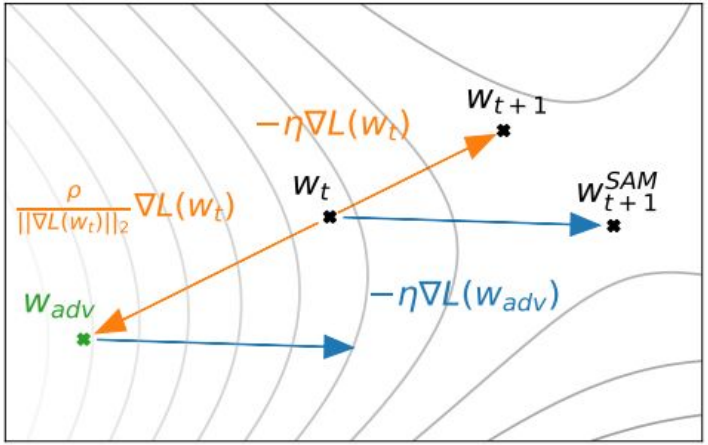
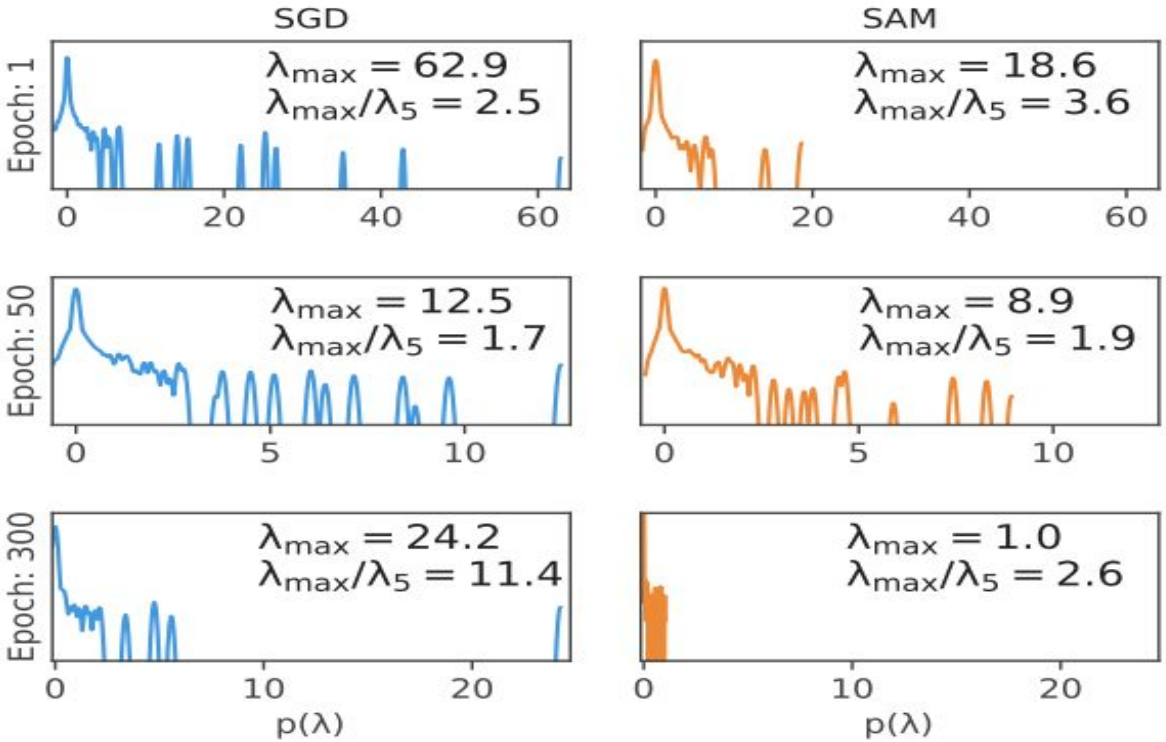


Figure 2: Schematic of the SAM parameter update.

Сравнение собственных значений Гессиан



		CIFAR-10		CIFAR-100	
Model	Augmentation	SAM	SGD	SAM	SGD
WRN-28-10 (200 epochs)	Basic	**2.7**$_{\pm 0.1}$	3.5$_{\pm 0.1}$	**16.5**$_{\pm 0.2}$	18.8$_{\pm 0.2}$
WRN-28-10 (200 epochs)	Cutout	**2.3**$_{\pm 0.1}$	2.6$_{\pm 0.1}$	**14.9**$_{\pm 0.2}$	16.9$_{\pm 0.1}$
WRN-28-10 (200 epochs)	AA	**2.1**$_{\pm <0.1}$	2.3$_{\pm 0.1}$	**13.6**$_{\pm 0.2}$	15.8$_{\pm 0.2}$
WRN-28-10 (1800 epochs)	Basic	**2.4**$_{\pm 0.1}$	3.5$_{\pm 0.1}$	**16.3**$_{\pm 0.2}$	19.1$_{\pm 0.1}$
WRN-28-10 (1800 epochs)	Cutout	**2.1**$_{\pm 0.1}$	2.7$_{\pm 0.1}$	**14.0**$_{\pm 0.1}$	17.4$_{\pm 0.1}$
WRN-28-10 (1800 epochs)	AA	**1.6**$_{\pm 0.1}$	2.2$_{\pm <0.1}$	**12.8**$_{\pm 0.2}$	16.1$_{\pm 0.2}$
Shake-Shake (26 2x96d)	Basic	**2.3**$_{\pm <0.1}$	2.7$_{\pm 0.1}$	**15.1**$_{\pm 0.1}$	17.0$_{\pm 0.1}$
Shake-Shake (26 2x96d)	Cutout	**2.0**$_{\pm <0.1}$	2.3$_{\pm 0.1}$	**14.2**$_{\pm 0.2}$	15.7$_{\pm 0.2}$
Shake-Shake (26 2x96d)	AA	**1.6**$_{\pm <0.1}$	1.9$_{\pm 0.1}$	**12.8**$_{\pm 0.1}$	14.1$_{\pm 0.2}$
PyramidNet	Basic	**2.7**$_{\pm 0.1}$	4.0$_{\pm 0.1}$	**14.6**$_{\pm 0.4}$	19.7$_{\pm 0.3}$
PyramidNet	Cutout	**1.9**$_{\pm 0.1}$	2.5$_{\pm 0.1}$	**12.6**$_{\pm 0.2}$	16.4$_{\pm 0.1}$
PyramidNet	AA	**1.6**$_{\pm 0.1}$	1.9$_{\pm 0.1}$	**11.6**$_{\pm 0.1}$	14.6$_{\pm 0.1}$
PyramidNet+ShakeDrop	Basic	**2.1**$_{\pm 0.1}$	2.5$_{\pm 0.1}$	**13.3**$_{\pm 0.2}$	14.5$_{\pm 0.1}$
PyramidNet+ShakeDrop	Cutout	**1.6**$_{\pm <0.1}$	1.9$_{\pm 0.1}$	**11.3**$_{\pm 0.1}$	11.8$_{\pm 0.2}$
PyramidNet+ShakeDrop	AA	**1.4**$_{\pm <0.1}$	1.6$_{\pm <0.1}$	**10.3**$_{\pm 0.1}$	10.6$_{\pm 0.1}$

ImageNet

Model	Epoch	SAM		Standard Training (No SAM)	
		Top-1	Top-5	Top-1	Top-5
ResNet-50	100	**22.5**$_{\pm 0.1}$	6.28$_{\pm 0.08}$	22.9$_{\pm 0.1}$	6.62$_{\pm 0.11}$
	200	**21.4**$_{\pm 0.1}$	5.82$_{\pm 0.03}$	22.3$_{\pm 0.1}$	6.37$_{\pm 0.04}$
	400	**20.9**$_{\pm 0.1}$	5.51$_{\pm 0.03}$	22.3$_{\pm 0.1}$	6.40$_{\pm 0.06}$
ResNet-101	100	**20.2**$_{\pm 0.1}$	5.12$_{\pm 0.03}$	21.2$_{\pm 0.1}$	5.66$_{\pm 0.05}$
	200	**19.4**$_{\pm 0.1}$	4.76$_{\pm 0.03}$	20.9$_{\pm 0.1}$	5.66$_{\pm 0.04}$
	400	**19.0**$_{\pm <0.01}$	4.65$_{\pm 0.05}$	22.3$_{\pm 0.1}$	6.41$_{\pm 0.06}$
ResNet-152	100	**19.2**$_{\pm <0.01}$	4.69$_{\pm 0.04}$	20.4$_{\pm <0.0}$	5.39$_{\pm 0.06}$
	200	**18.5**$_{\pm 0.1}$	4.37$_{\pm 0.03}$	20.3$_{\pm 0.2}$	5.39$_{\pm 0.07}$
	400	**18.4**$_{\pm <0.01}$	4.35$_{\pm 0.04}$	20.9$_{\pm <0.0}$	5.84$_{\pm 0.07}$

Table 2: Test error rates for ResNets trained on ImageNet, with and without SAM.

SWA (Stochastic Weight Averaging)

1. Вдохновимся идеей FGE и будем усреднять веса вдоль кривых между локальными минимумами

Преимущества:

1. Всего одна модель
2. Решает проблему большого инференса

Loss landscape

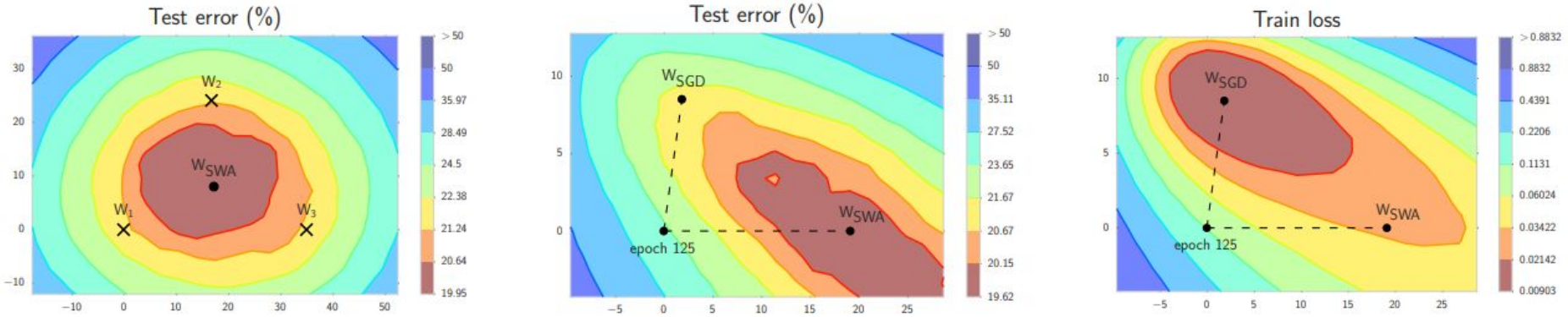
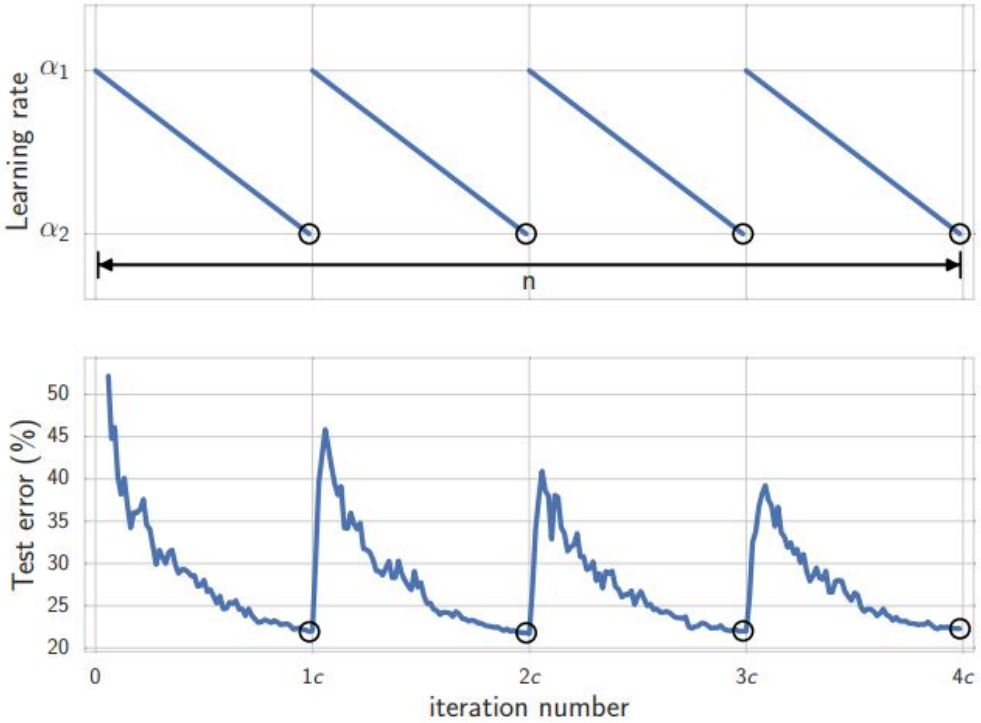


Figure 1: Illustrations of SWA and SGD with a Preactivation ResNet-164 on CIFAR-100[1]. **Left**: test error surface for three FGE samples and the corresponding SWA solution (averaging in weight space). **Middle** and **Right**: test error and train loss surfaces showing the weights proposed by SGD (at convergence) and SWA, starting from the same initialization of SGD after 125 training epochs.

Learning rate



$$\alpha(i) = (1 - t(i))\alpha_1 + t(i)\alpha_2,$$

$$t(i) = \frac{1}{c}\left(\mathrm{mod}(i-1, c) + 1\right).$$

Зависимость ошибки от итерации модели ResNet164 на датасете CIFAR-100

Реализация метода 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 $\mathrm{mod}(i, c) = 0$ **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}

Связь с широким оптимумом

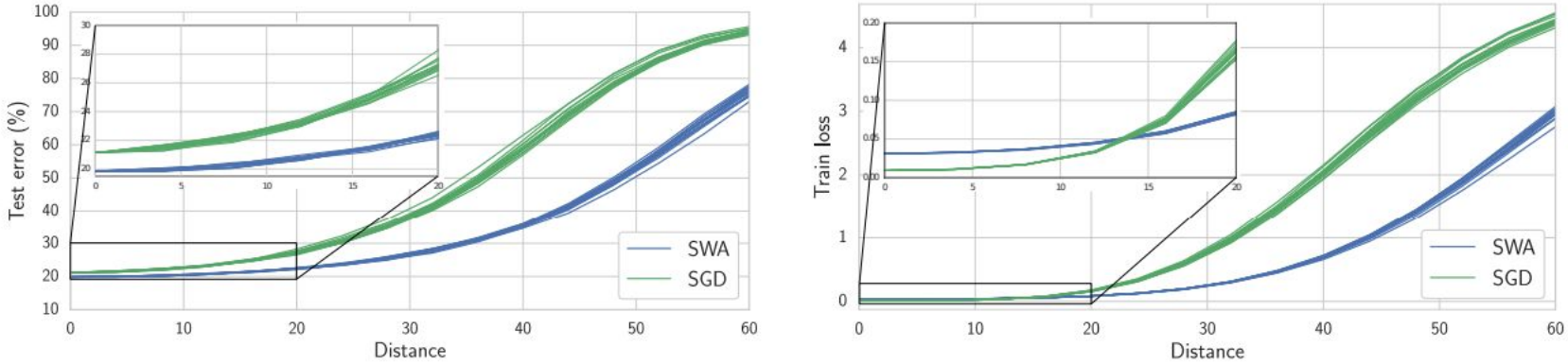


Figure 4: (**Left**) Test error and (**Right**) L_2-regularized cross-entropy train loss as a function of a point on a random ray starting at SWA (blue) and SGD (green) solutions for Preactivation ResNet-164 on CIFAR-100. Each line corresponds to a different random ray.

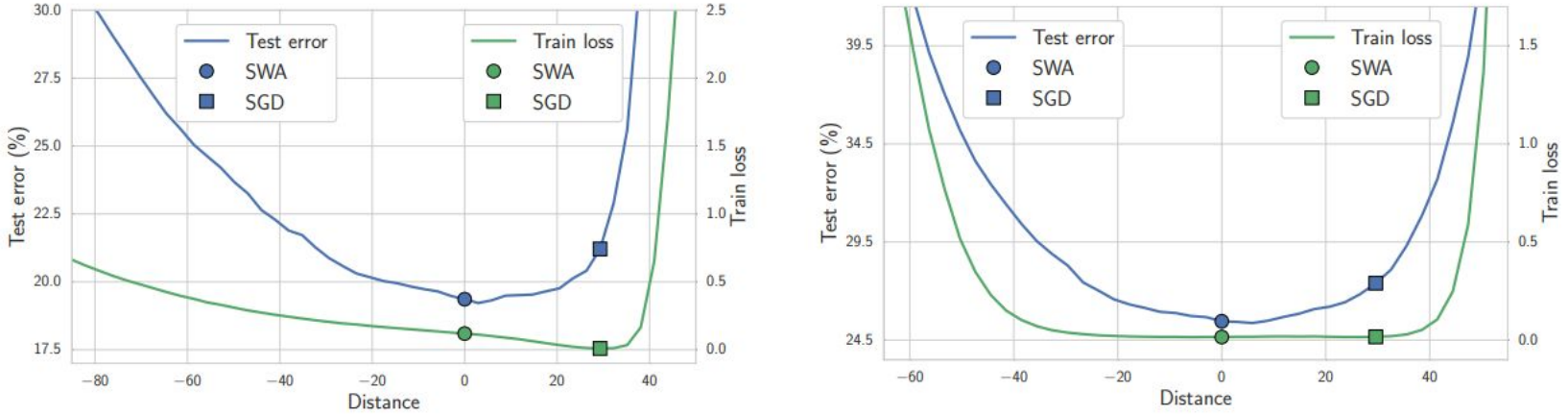


Figure 5: L_2-regularized cross-entropy train loss and test error as a function of a point on the line connecting SWA and SGD solutions on CIFAR-100. **Left**: Preactivation ResNet-164. **Right**: VGG-16.

DNN (Budget)	SGD	FGE (1 Budget)	SWA		
			1 Budget	1.25 Budgets	1.5 Budgets
		CIFAR-100			
VGG-16 (200)	72.55 ± 0.10	74.26	73.91 ± 0.12	74.17 ± 0.15	74.27 ± 0.25
ResNet-164 (150)	78.49 ± 0.36	79.84	79.77 ± 0.17	80.18 ± 0.23	80.35 ± 0.16
WRN-28-10 (200)	80.82 ± 0.23	82.27	81.46 ± 0.23	81.91 ± 0.27	82.15 ± 0.27
PyramidNet-272 (300)	83.41 ± 0.21	–	–	83.93 ± 0.18	84.16 ± 0.15
		CIFAR-10			
VGG-16 (200)	93.25 ± 0.16	93.52	93.59 ± 0.16	93.70 ± 0.22	93.64 ± 0.18
ResNet-164 (150)	95.28 ± 0.10	95.45	95.56 ± 0.11	95.77 ± 0.04	95.83 ± 0.03
WRN-28-10 (200)	96.18 ± 0.11	96.36	96.45 ± 0.11	96.64 ± 0.08	96.79 ± 0.05
ShakeShake-2x64d (1800)	96.93 ± 0.10	–	–	97.16 ± 0.10	97.12 ± 0.06

Table 1: Accuracies (%) of SWA, SGD and FGE methods on CIFAR-100 and CIFAR-10 datasets for different training budgets. Accuracies for the FGE ensemble are from Garipov et al. [2018].

Связь SWA и FGE

$$w_{SWA} = \frac{1}{n}\sum w_i,\ w_{FGE} = \frac{1}{n}\sum f(w_i)$$

Пусть $\Delta_i = w_i - w_{SWA}$, тогда $\sum \Delta_i = 0$
Лианезируем f в точке w_{SWA}

$$f(w_j) = f(w_{SWA}) + < \nabla f(w_{SWA}), \Delta_j > + O\left(\|\Delta_j\|^2\right)$$

тогда

$$\bar{f} - f(w_{\text{SWA}}) = \frac{1}{n}\sum_{i=1}^{n}\left(\langle \nabla f(w_{\text{SWA}}), \Delta_i\rangle + O\left(\|\Delta_i\|^2\right)\right) = \left\langle \nabla f(w_{\text{SWA}}), \frac{1}{n}\sum_{i=1}^{n}\Delta_i\right\rangle + O(\Delta^2) = O(\Delta^2),$$

Преимущества триангулярного lr

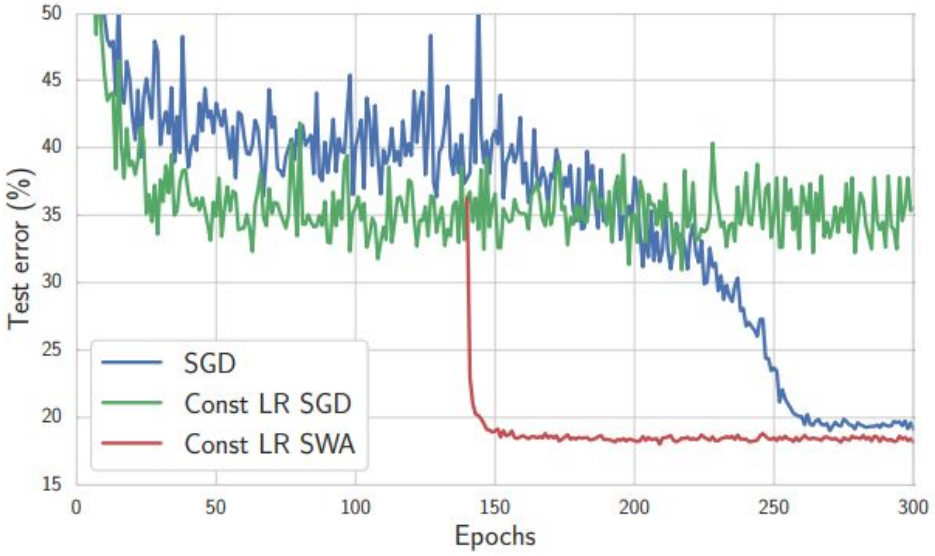


Figure 7: Test error as a function of training epoch for constant (green) and decaying (blue) learning rate schedules for a Wide ResNet-28-10 on CIFAR-100. In red we average the points along the trajectory of SGD with constant learning rate starting at epoch 140.

Используемые ресурсы

1. Статья про SAM
2. Статья про SWA