1. SGD optimizer - sgd-50-epoch.PNG
2. Adam optimizer 100 epochs, no weight decay(acc) – adam-noweight-decay-acc.PNG
3. Adam optimizer 100 epochs, no weight decay (loss) – adam-no-weight-decay.png
4. Adam optimizer 100 epochs, no alpha scheduler(acc) - adam-100-acc-no-scheduler.png
5. Adam optimizer 100 epochs, no alpha scheduler(loss) - adam-100-loss-no-scheduler.png
6. Adam optimizer 100 epochs, alpha scheduler at 50 epochs (loss) – adam-100.png
7. Adam optimizer 100 epochs, alpha scheduler at 50 epochs(acc) – adam-100-acc.png
8. Adam optim(acc) – adam-optim-acc.png
9. Adam optim(loss) – adam-optim-loss.png

Alpha = 0.001, alpha decay 0.1

Weight\_decay = 0.0001

Momentum 0.9

Less epoch see point (1)

Batch size

The issue is that a small batch size both helps and hurts convergence. **Updating the weights based on a small batch will be more noisy**. The noise can be good, helping by jerking out of local optima.

The issue is that a small batch size both helps and hurts convergence. **Updating the weights based on a small batch will be more noisy**. The noise can be good, helping by jerking out of local optima.

Option A vs B

Next

we investigate projection shortcuts (Eqn.(2)). In Table 3 we

compare three options: (A) zero-padding shortcuts are used

for increasing dimensions, and all shortcuts are parameterfree

(the same as Table 2 and Fig. 4 right); (B) projection

shortcuts are used for increasing dimensions, and other

shortcuts are identity; and (C) all shortcuts are projections.

Table 3 shows that all three options are considerably better

than the plain counterpart. B is slightly better than A.We

argue that this is because the zero-padded dimensions in A

indeed have no residual learning. C is marginally better than

B, and we attribute this to the extra parameters introduced

by many (thirteen) projection shortcuts. But the small differences

among A/B/C indicate that projection shortcuts are

not essential for addressing the degradation problem. So we

do not use option C in the rest of this paper, to reduce memory/

time complexity and model sizes. Identity shortcuts are

particularly important for not increasing the complexity of

the bottleneck architectures that are introduced below.