進捗報告

表 1: 実験の設定

base model	VGG19
Optim(w)	SGD(lr=0.0090131, momentum=0.9)
Scheduler(w)	Step(γ=0.2344, stepsize=100)
Loss	Cross Entropy Loss
dataset	cifar10
batch size	64
epoch	150

1　今週やったこと

- ランダムアーキテクチャの評価実験
- GA の実装準備
- 事前学習重みでのアーキテクチャ探索

2　評価実験

ランダムアーキテクチャを複数回実験してベースラインを確認する.

表 1 に評価時の実験設定を示した.

2.1　結果

表 2 にはテスト精度の結果を示した. 図 1, 2 にはパラメータ数, ショートカット数に対するそれぞれの精度を図示した. 精度や傾き (モデルスケールに対する精度) から手法 B が優れているとする.

3　GA の準備

deap でサンプルの GA を動作を確認.
問題

- 個体表現 (α は行列)
- 交叉・突然変異の方法
- メモリに乗るか?
- 実行時間

とりあえず今の DARTS のコードに実装してみる.

4　事前学習実験

DARTS は w を近似しているので高速である. ベースモデルを VGG には ImageNet の事前学習モデルがある. この重み w を利用するとより高性能なアーキテクチャが得られる？

4.1　探索

試しに 1 回実験した. 図 3 には探索中のテスト精度を示した. これによると初期段階から高い精度が得られているため, ファインチューニングできている.

図 4, 5, 6 にはファインチューニングによる具体的なアーキテクチャの違いを図示した. 図 7, 8 には探索の結果の α を示した. 図 8 では, ファインチューニングによってより深い層でショートカットが必要になっている.

図 5, 6 では 4 ブロック以上離れた位置のショートカット数が多い. 手法 A より B の方が性能が高かったため, 6 はさらに高い性能となることが期待される.

4.2　評価

usagi サーバーで実験を回そうとするとエラーを吐いてからログインできなくなった??

5　今後の予定

- 評価実験回す
- できれば GA の実装

6　ソースコード

github の notebook リポジトリ参照.

表 2: 各アーキテクチャの精度

architecture		**test accuracy (%)**	**param (M)**	**number of shortcuts**	**random architect accuracy (%)**
architecture search A	50 epoch	93.70 ± 0.22	21.06 ± 0.07	12.7 ± 1.4	93.60 ± 0.15
	100 epoch	94.02 ± 0.12	21.50 ± 0.11	18.2 ± 0.9	93.67 ± 0.14
	150 epoch	93.90 ± 0.17	21.57 ± 0.25	18.9 ± 0.6	93.64 ± 0.09
architecture search B	50 epoch	93.57 ± 0.19	20.45 ± 0.09	5.8 ± 1.2	93.36 ± 0.19
	100 epoch	93.93 ± 0.08	20.73 ± 0.10	9.8 ± 1.0	93.47 ± 0.17
	150 epoch	93.92 ± 0.12	20.76 ± 0.15	10.6 ± 1.0	
baseline (VGG19)		93.03 ± 0.10	20.04	0	-

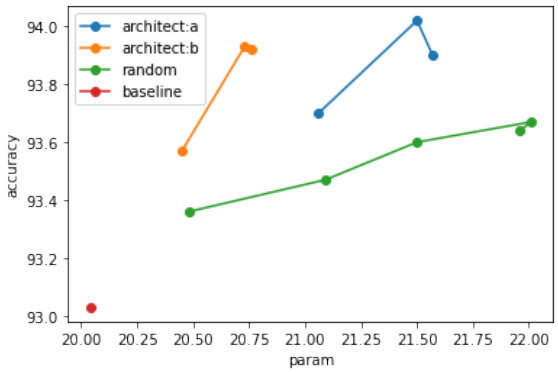

図 1: パラメータ数に対する精度

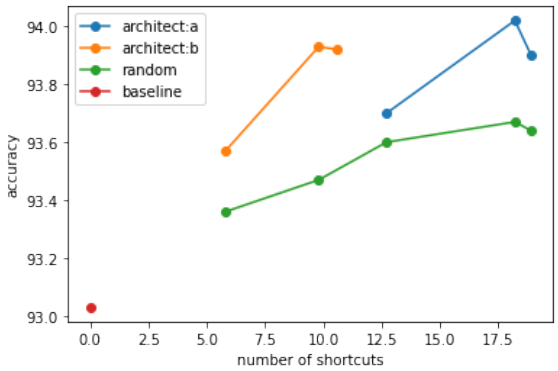

図 2: ショートカット数に対する精度

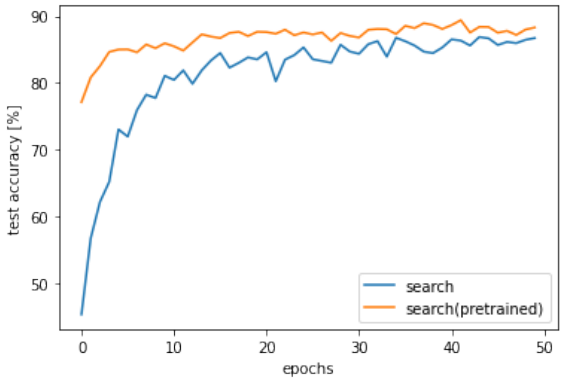

図 3: w を事前学習した探索時のテスト精度

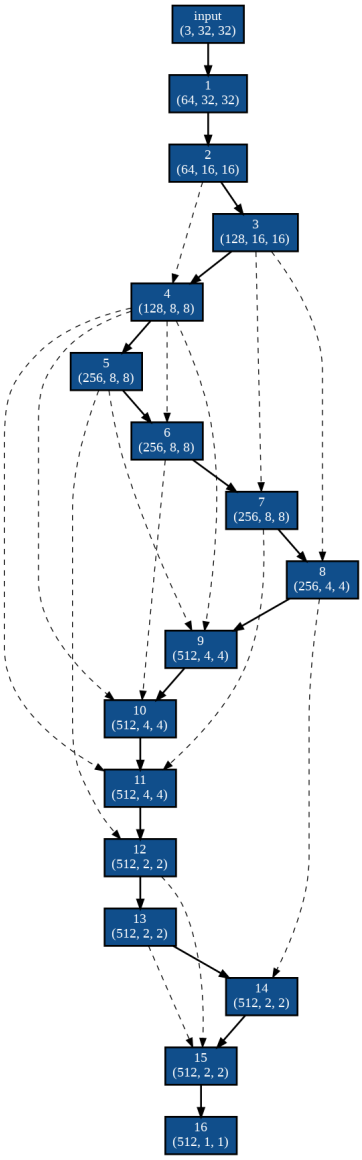

図 4: 手法 A のグラフ (50 epoch)

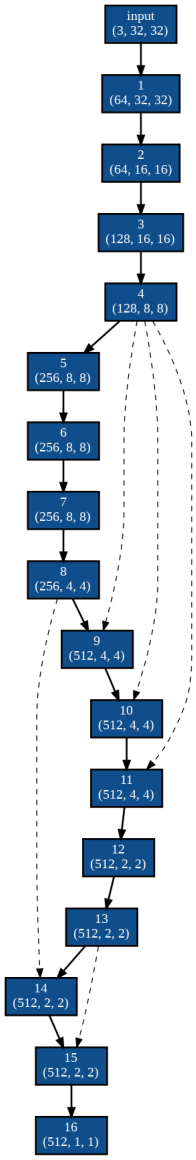

図 5: 手法 B のグラフ (50 epoch)

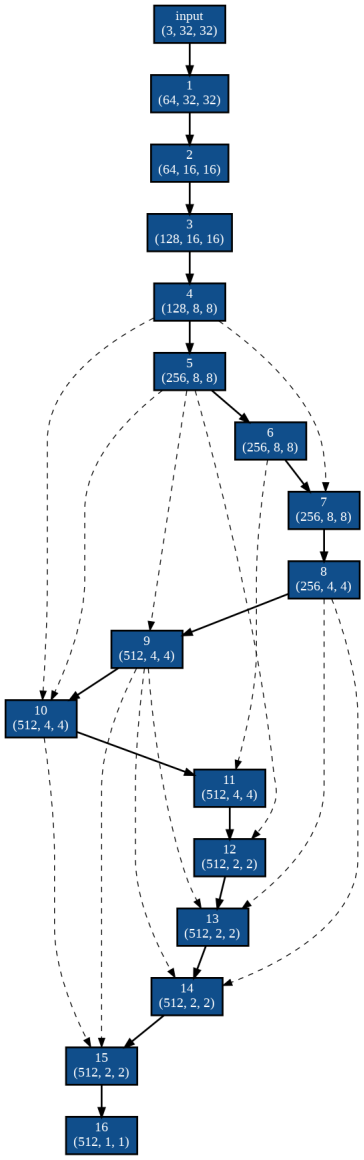

図 6: 手法 A のグラフ (50 epoch, pre-trained)

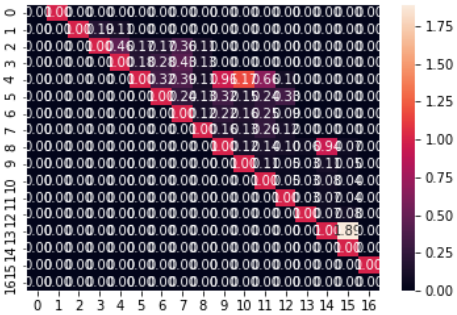

図 7: 事前学習なしの α(従来)

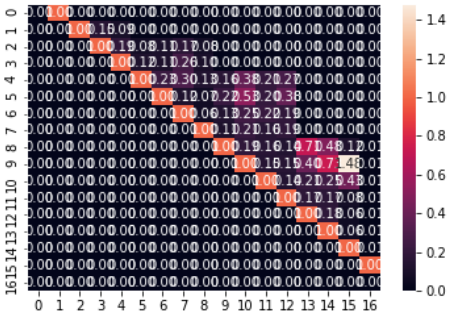

図 8: 事前学習ありの α(今回)