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

- 閾値によるアーキテクチャの評価実験
- 猫データセットへの適用

2　実験

前回までの結果 (A) に加え, 閾値によるアーキテクチャ決定をした場合 (B) の性能評価を行った.

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. A, B では探索結果の α が同じ条件で, 手法の違いを比較した.

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. この結果から 150 epoch 時点で, 2 つの手法で性能はともに下がっているため手法に拠らず, 探索の結果の α 自体が悪くなっている可能性が高い. また手法 B では性能は劣るものの, アーキテクチャのスケールに対して効率の良い精度が得られていることが分かった. 手法 A がより優れた性能であるのは, 単にパラメータ数が多いからであると思われる.

3　考察

ランダムアーキテクチャの試行回数を増やして, 図 1, 2 にプロットすれば, 探索の効果がより分かりやすくなると思われる.

VGG19 の事前学習モデルを利用することで近似の精度が上がり, より有望な α の探索ができると考えた. ファインチューニングの動作確認もできたので,

- 通常の探索時の初期重み
- GA での探索時の初期重み

などの実験に使いたい.

4　実験:猫

224x224x3 の猫データセット 500 枚を共有してもらったので, CIFAR10 以外のデータセットへの適用を考えた.

4.1　問題

Google Colab で開発していて水曜日までは動作していたコードが木曜日に動かなくなった?. (内部の環境がアップデートされた?) バグの修正方法を調査中.

5　今後の予定

- バグの修正
- できれば GA の実装
- 論文の調査

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
architecture A	50 epoch	93.70 ± 0.22	21.06 ± 0.07	12.7 ± 1.4
	100 epoch	94.02 ± 0.12	21.50 ± 0.11	18.2 ± 0.9
	150 epoch	93.90 ± 0.17	21.57 ± 0.25	18.9 ± 0.6
architecture B	50 epoch	93.57 ± 0.19	20.45 ± 0.09	5.8 ± 1.2
	100 epoch	93.93 ± 0.08	20.73 ± 0.10	9.8 ± 1.0
	150 epoch	93.92 ± 0.12	20.76 ± 0.15	10.6 ± 1.0
random architect		93.60 ± 0.15	21.50 ± 0.23	12.7 ± 1.4
baseline (VGG19)		93.03 ± 0.10	20.04	0

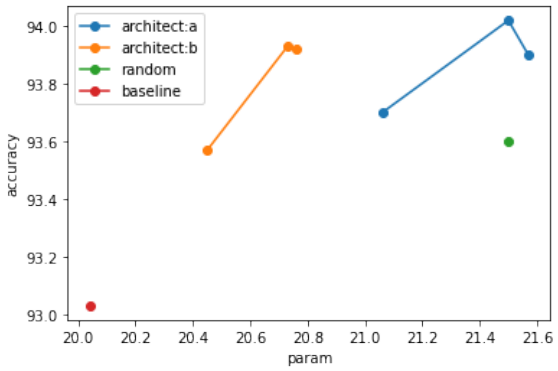

図 1: パラメータ数に対する精度

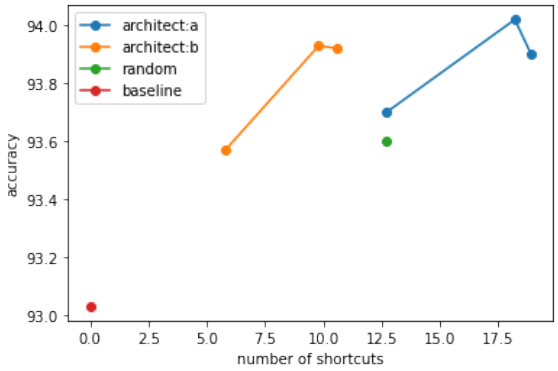

図 2: ショートカット数に対する精度

Listing 1: error code

```
def normalized_alpha(self):
    alpha = torch.zeros_like(self.alphas[0])
    if self.evaluate:
      ...
    else:
      alpha[self.mask_f] = 1.0
      for a, raw, mask, b in zip(alpha, self.alpha(), self.mask_s, self.normalized_beta()):
        a[mask] = b * F.softmax(raw[mask], dim=0)
    return alpha

RuntimeError: Output 0 of a function created in no_grad mode is a view and is being modified
    inplace but this inplace operation is not allowed. You should either replace it by an out of
     place operation or do a .clone() of the Tensor before modifying it inplace. Note that this
    can happen when using DataParallel or DistributedDataParallel, which can send views of the
    original input to the forward method of the model.
```