進捗報告

1　今週やったこと

- 後述するデータを交換するアルゴリズムの実装と実験

2　問題設定

データを正解と不正解に分割して学習する以下のアルゴリズム 1 を考える.

Algorithm 1 Swap two datasets

1. データ 2N 抜き出す
2. そのデータを N（A） と N（B）に分ける.
3. A → train, B → test で実験
4. B → train, A → test で実験
5. 3. と 4. で test でミスしたデータを調べる
6. A をテストとしたとき失敗したデータと B をテストとしたとき成功したデータを入れ替える.
7. 3. に戻る.

アルゴリズム 1 の手順 6 の操作によって, データセット A には成功データ, B には失敗データが集められる. 従って簡単なデータを学習し, 難しいデータでテストされる手順 3 の精度は低くなると予想される.

アルゴリズム 1 に示した手順の設定として, 手順 3. 4. で 2 つのモデルをそれぞれ独立させた. またデータポイントの入れ替えは, 入れ替え可能なデータを先頭から全て交換することにした.

3　実験

実験 1 アルゴリズム 1 で学習

実験 2 データ交換の実装をテストするため, 手順 3. 4. で訓練を行わずアルゴリズム 1 を学習

実験 3 ベースラインとしてデータ数をNのまま, データを入れ替えず学習

表 1: 実験の設定

dataset	cifar10
data N	25,000 / model
task	10 クラス識別
input	image(3x32x32)
output	class(10)
model	CNN
optim	SDG (lr=0.001, moment=0.9)
loss	Cross Entropy Loss
batch size	16
epoch	25

表 2: 実験 2 の手順 6 の結果

テスト	結果 (o:正解, x:不正解)	交換
A	x x x x x x x o x o	1 番目
B	x x o x x x x x x x	3 番目

3.1　共通実験設定

実験 1 ～ 3 は, 条件にない限り表 1 の設定に従って学習した. データセットに CIFAR10[1] を利用したため, モデルに与える問題は 10 クラス識別とした. 入力は 3 channel x 32 pix x 32 pix の画像で, 出力は 10 クラスの確率である. 多クラス識別のため損失関数は Cross Entropy Loss を利用した. 手順 3. 4. の実験の期間は, 1 epoch と設定して, 25 epoch で 25 回の入れ替えを行った.

3.2　実験結果

3.2.1　実装の確認

実験 2 での入れ替え例を表 2 に示した. 一部抜き出した 10 件のデータで, 先頭から期待通りに交換されていることを確認した. また全体として, 図 1 のテストの精度から, 1 epoch 目でほとんどのデータが交換されたことが分かる. 以上より実装に問題がないと判断した.

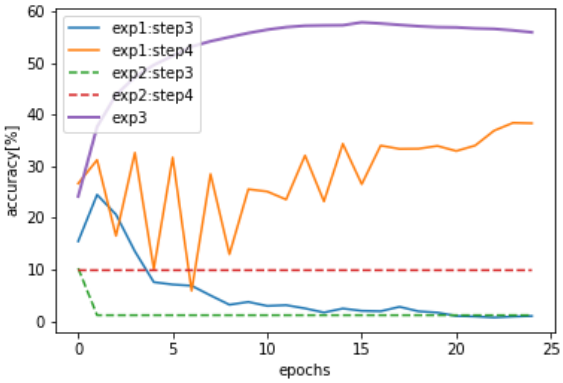

図 1: テストの精度　(実験 1) 実線の青が手順 3 で 0 に近い精度, 橙が手順 4 で不安定に向上. (実験 2) 点線の緑が手順 3, 赤が手順 4 でベースラインに近い. (実験 3) 太線の紫, 最も高い精度.

3.2.2　実験 1 の結果と考察

図 1 のように予想通り手順 3 は学習を重ねるごとに, 精度が下がり 0.992 [%] になった. 反対に手順 4 は学習が進み, 精度は 38.3 [%] となった. 一方で実験 3 の通常の学習では同数のデータセットかつ同一条件で, 最も高い 56.0 [%] となった. 実験 3 と比較して 20 [%] 程度精度が劣った理由は, データセットが頻繁に操作され不安定であることや, データの多様性が失われたことによる汎化能力の低下が考えられる.

図 1 の手順 4 のグラフで精度が交互に激しく変化する現象が見られた. これデータセットが入れ替えによって, 難易度が大きく変化しテストに影響していると思われる. しかしデータの流れやテストの難易度は分かっていないため, 調査が必要である.

4　現在の問題点

現在の実装では, 入れ替えを先頭から全て行うように設定している. 偏りをなくすためランダムに選択したり, 一部のみを入れ替えられる必要がある.

ベースラインの CNN の精度が高くなかったため, CNN を改良し結果を比較したい.

精度の変化を説明するため, 入れ替えたデータの量を測りデータの流れを分析したい.

入れ替える実験の期間として, 1 epoch が適切かどうかが不明のため, 最適な期間を探す実験が必要な可能性がある.

実装について, CPU で行っているデータの入れ替えを GPU で再実装し, 高速化したい.

5　現在のソースコード

モデルの実装は Github [1] を参照.

6　今後の予定

- 具体的なデータを分析し, 精度の妥当性について考察する
- Pytorch を調べて, GPU による演算を実装し, 学習時間の短縮を行う

参考文献

[1] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[1] https://github.com/tatsuya-sugiyama/WeeklyReport/blob/master/test/AB_split.ipynb