進捗報告

1　今週行ったこと

先週と同様に、Googlecolab を用いて、Cifar10 の CNN の以下の実装を行ったが、削除されたデータではどのように識別率が変わっていくのかを実験した。前回の結果より、40 回の試行でデータが 2 クラスほどしか残らなかったので、他の初期条件でも同様になるか確かめた。

1. データを 10000 個取り出す
2. データを A と B にランダムに二分割する
3. A→train,B→test で実験（A→train,B→test としたとき AtoB と表記する）
4. B→train,A→test で実験（B→train,A→test としたとき BtoA と表記する）
5. 3. と 4. でミスしたデータを調べる
6. A をテストした時、失敗したデータ　B をテストした時、失敗したデータをランダムに M 個ずつ削除（自分で指定）
7. 3.-6. を二回繰り返す
8. 二回の繰り返しで得られた難しいデータ (削除された側のデータ) に対して 3.-6. を繰り返す

上記のアルゴリズムより、前回の結果と比較して、データ数はもっと少なくなると予想される。モデルの概要は以下である。

クラス	10 クラス分類（Cifar10）
データ数	10000
input	image(32 × 32 × 3)
output	class(10)
モデル	CNN
optimizer	Adam
損失関数	categorical_crossentropy

以下に、変数を定義する

- random: A,B のシャッフルの random_state
- lr:学習率
- batch:　バッチサイズ
- epoch: 3.4 での訓練回数（エポック数）
- delete_num: 1 回の削除で削除するデータの最大数
- iteration: 3.4.5.6. の繰り返し数

パラメータは random=0,lr=0.001,batch=128,epoch=25,delete_num=10000,iteration=40 とした。

また、各クラスの数は図 1、図 2 のようになった。残るデータにはある程度の偏りが見られたが、残ったデータが動物クラス、機械クラスのどちらかに偏ることはなかった。片方に残っているデータはどちらかの識別が 100 %になっているためであるので両方のクラスに残っているものに注目すると以下の図のようになった。

残ったラベル

乱数	両方	A	B
0	1,2	0,1,2,3	1,2
1	1,4,8	1,4,8	1,3,4,5,8
2	3,9	3,5,8,9	3,9
3	0,3,4	0,3,4	0,3,4
4	0,4,9	0,4,9	0,4,5,7,9
5	0,2,3,9	0,2,3,9	0,2,3,4,5,7,9
6	0,3,4	0,3,4	0,3,4
7	3,8	3,4,8	3,8
8	0,4	0,1,4	0,4
9	0,4	0,3,4	0,4

両方ともに残ったラベルの回数

ラベル	0	1	2	3	4	5	6	7	8	9
ラベル名	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
回数	6	2	2	5	6	0	0	0	2	3

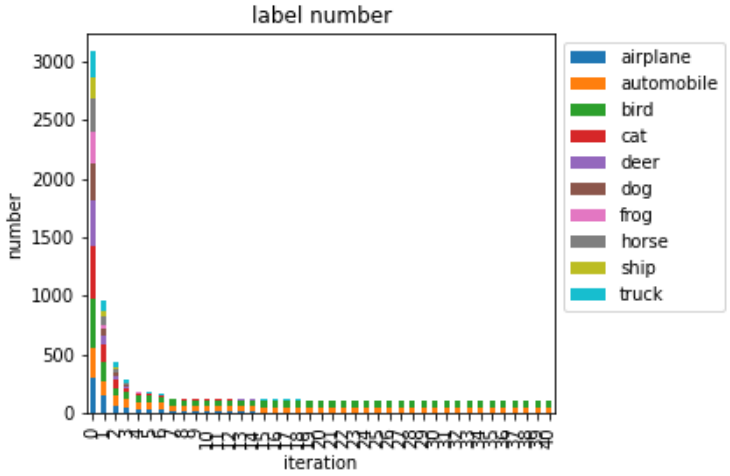

図 1: random=0 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

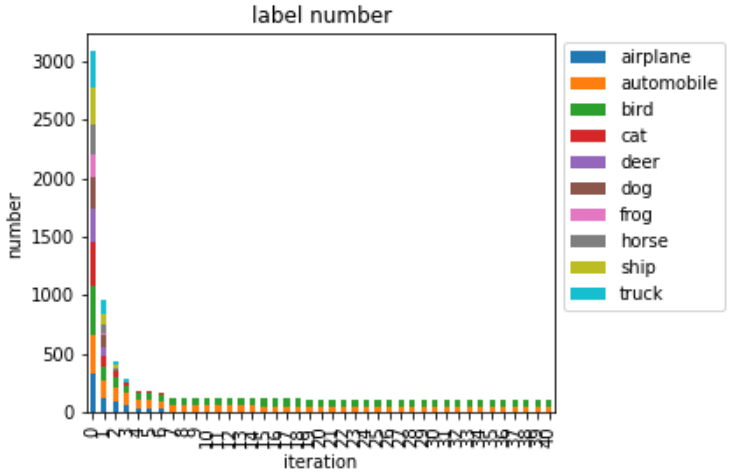

図 2: random=0 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

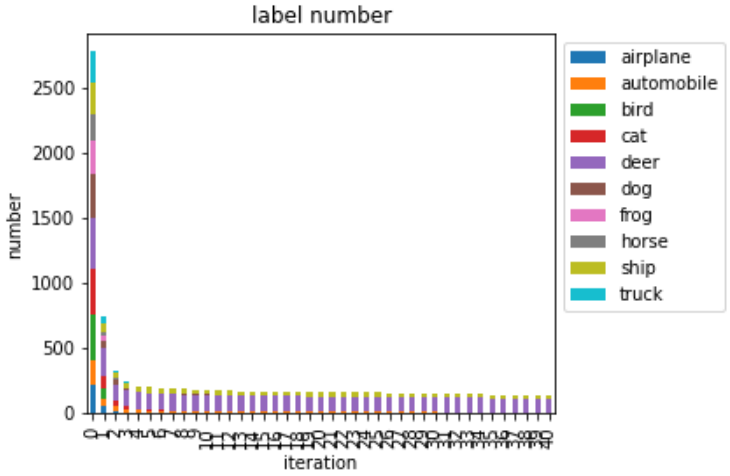

図 3: random=1 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

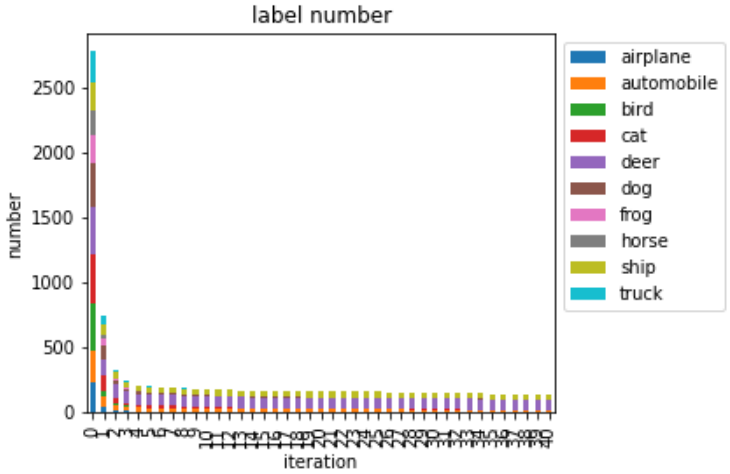

図 4: random=1 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

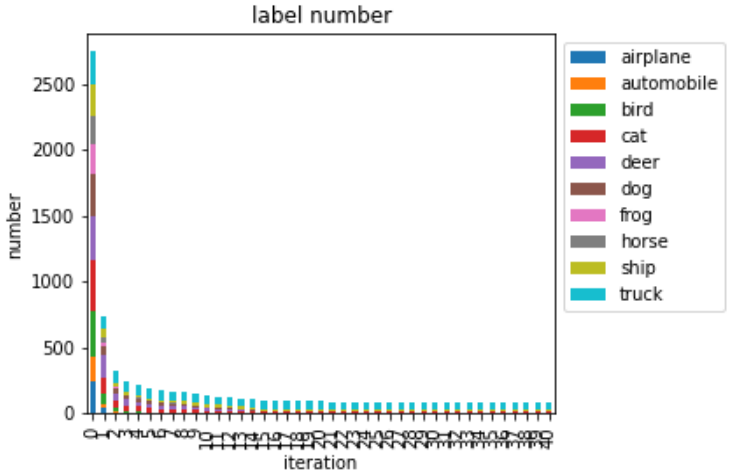

図 5: random=2 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

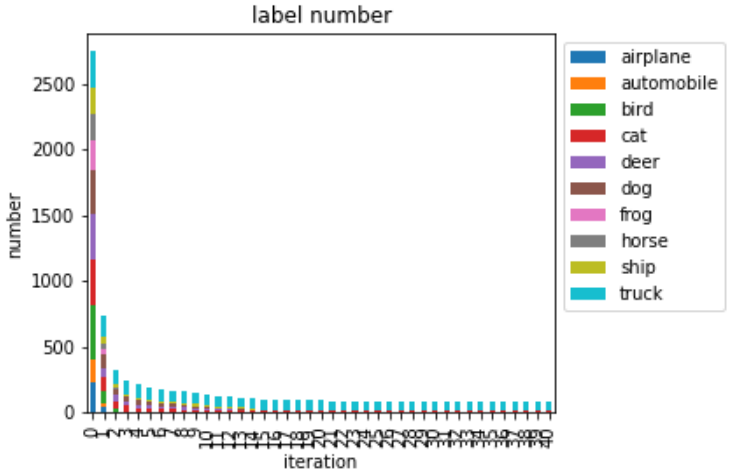

図 6: random=2 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

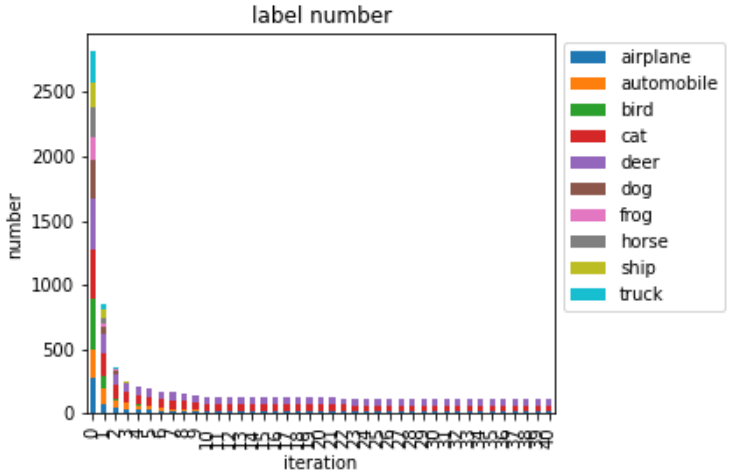

図 7: random=3 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

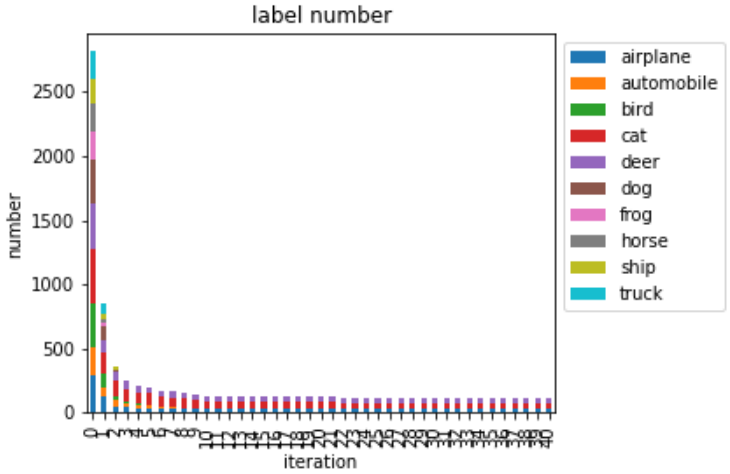

図 8: random=3 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

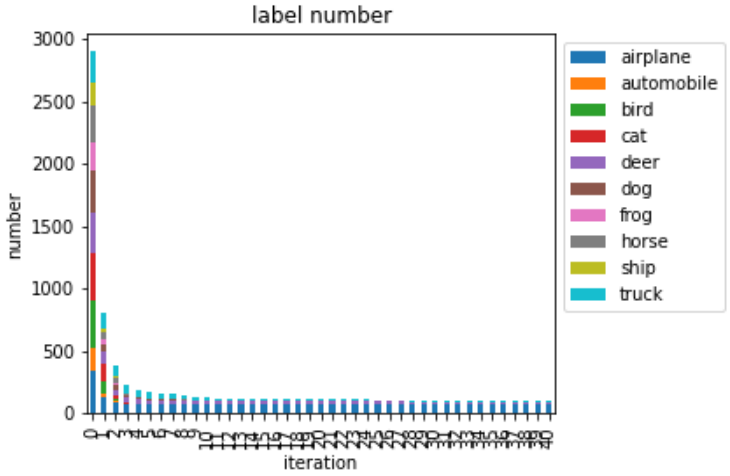

図 9: random=4 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

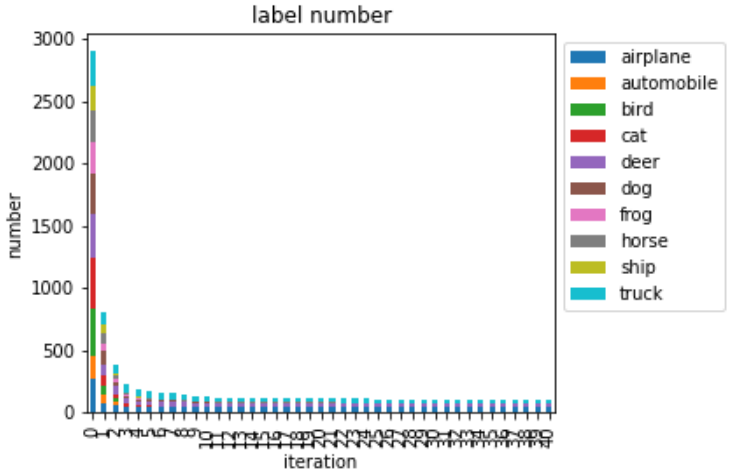

図 10: random=4 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

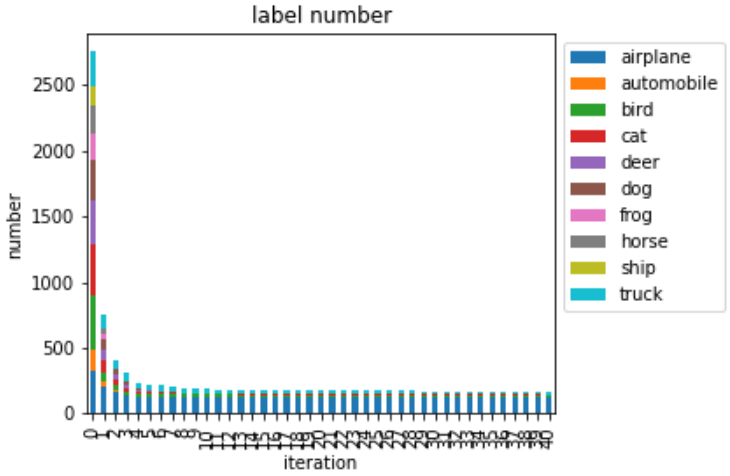

図 11: random=5 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

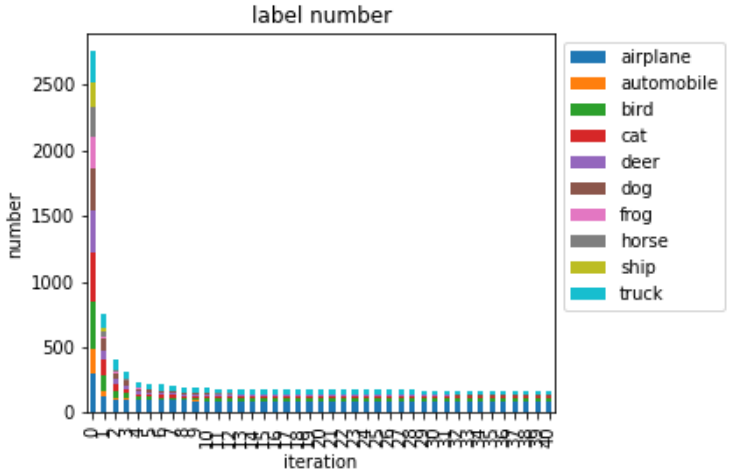

図 12: random=5 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

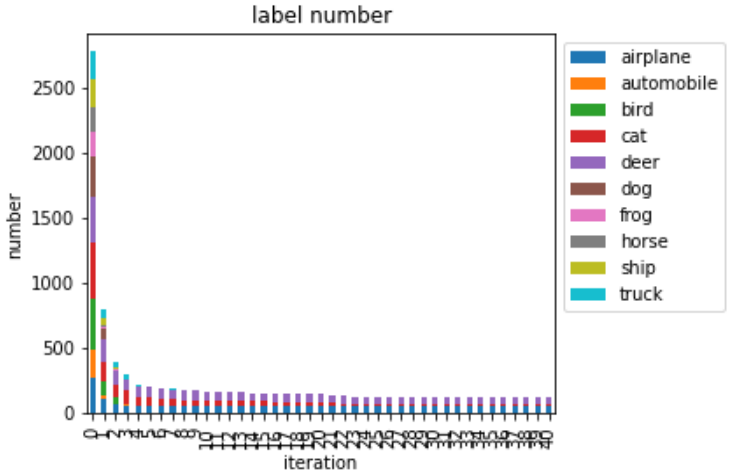

図 13: random=6 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

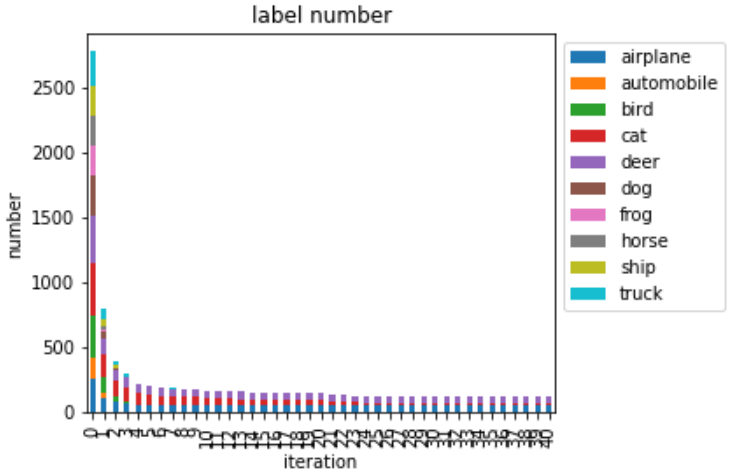

図 14: random=6 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

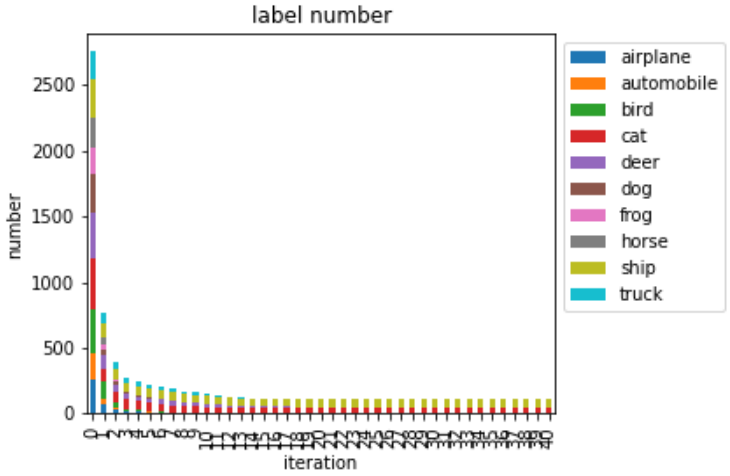

図 15: random=7 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

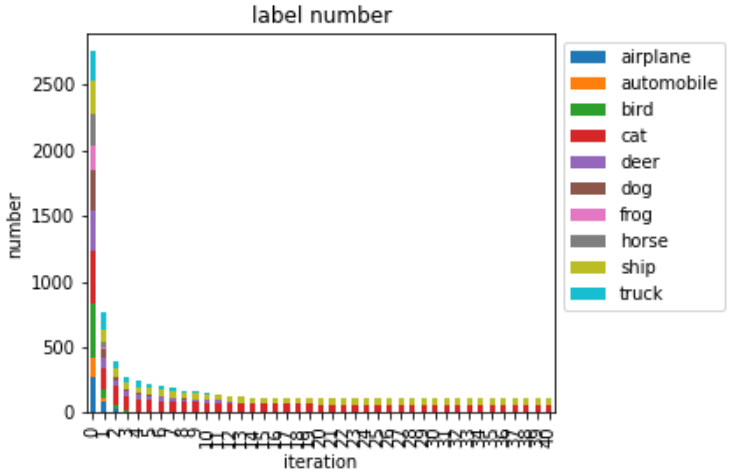

図 16: random=7 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

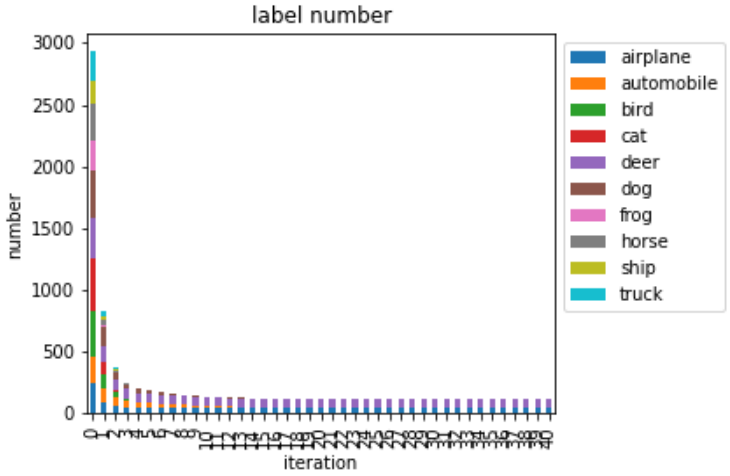

図 17: random=8 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

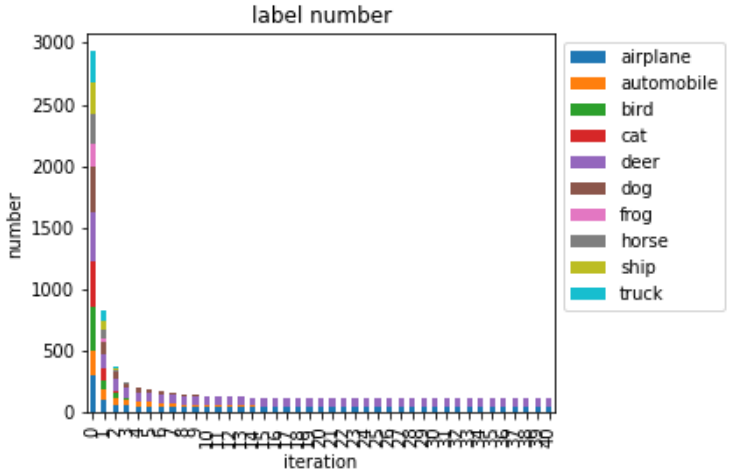

図 18: random=8 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

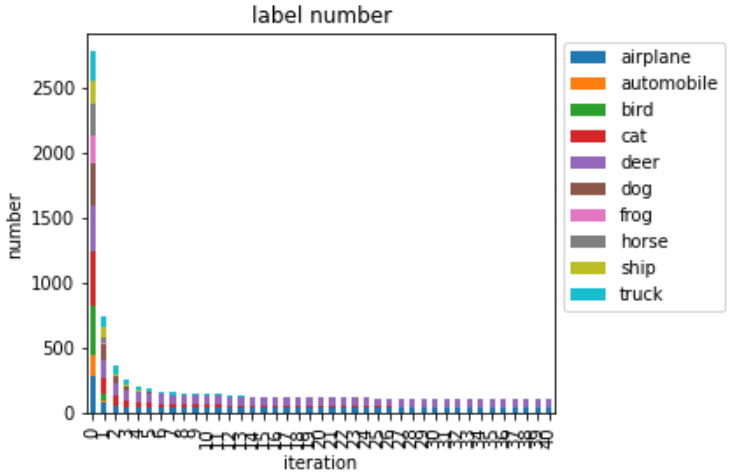

図 19: random=9 の A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

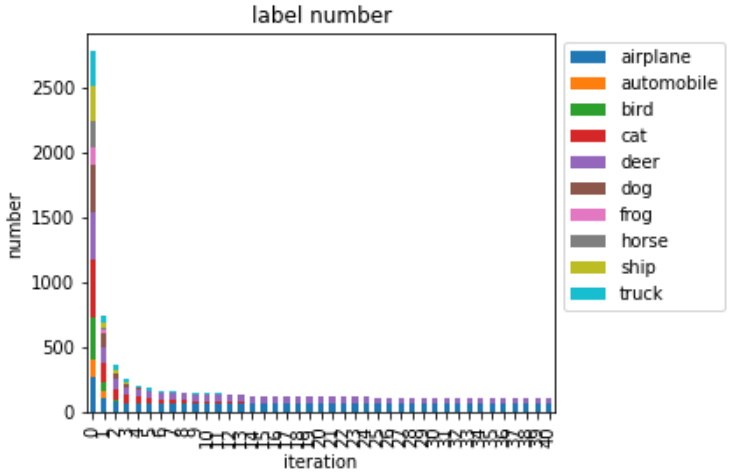

図 20: random=9 の B のラベルの数の推移。横軸が繰り返す回数、縦軸が B のラベルの数。

1. Cifar10のデータ10000件取り出す
2. 取り出したデータをA,Bに二分割する

モデルのイメージ

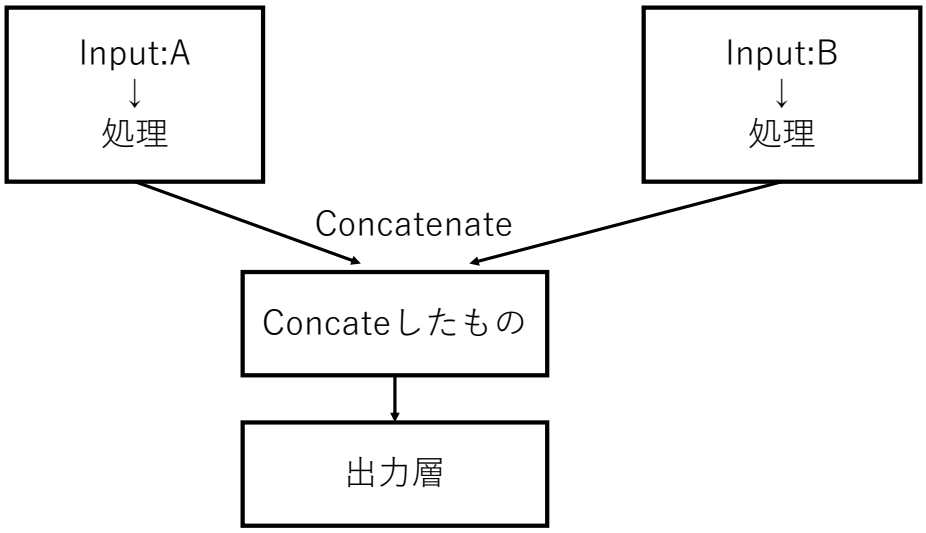

図 21: A のラベルの数の推移。横軸が繰り返す回数、縦軸が A のラベルの数。

2 考察

残るデータにはある程度の偏りが見られたが、残ったデータが動物クラス、機械クラスのどちらかに偏ることはなかった。もう少し試行回数を増やしてみたい。

3 次回行うこと

- concat モデルの実装

現時点でのコードを以下に示す。
https://github.com/KeitaTakami/WeeklyReport/blob/master/0605/cifar10_missdata.ipynb

実験に使った CNN のモデルを下記に示す.

Listing 1: model

```
1
2 model=Sequential()
3
4 #1st layer
5 model.add(Conv2D(32,kernel_size=(3,3),activation='relu',padding='same',input_shape=(32,32,3)))
6 model.add(Conv2D(32,kernel_size=(3,3),activation='relu'))
7 model.add(MaxPooling2D(pool_size=(2,2)))
8 model.add(Dropout(0.25))
9
```

```
#2nd layer
model.add(Conv2D(32,kernel_size=(3,3),activation='relu',padding='same'))
model.add(Conv2D(32,kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

#Output layer
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10))
model.add(Activation('softmax'))
```