Machine Learning hw3 Semi-Supervised Learning

物理三 B03202017 李漪莛

1. Supervised

Method

- 訓練時取出 label 中 500~1000 筆作為 validation 測試，若繳交結果至 kaggle 時則不分離 validation，並且 x_label, y_label 要事先同步 shuffle 以利取到各類別資料。
- Model（regularization 使用 l2、lam=0.03）

  Convolution (25, 3, 3) -> MaxPooling (2, 2) -> Convolution (50, 3, 3) -> MaxPooling (2, 2) -> Flatten -> Dense (500) -> dropout (0.4) -> Sigmoid -> Dense (10) -> Softmax

Performance

- 若超過三組 Convolution + MaxPooling 效果不會顯著進步，但訓練時間會長許多。
- 使用 1~2 層 Dense layer 效果較好，若超過會 overfitting（train acc 到 1，但 validation 約只有 0.4~0.45）。而 1 的 validation acc 和 train acc 差距又較 2 層小。
- 不同 layer 數及不同 dropout 在 epoch = 40 結果比較（acc 可能為數次平均）

編號	layer 數	neuron 數	dropout	train acc	validation acc
A	1	128	0	約 0.96~0.98	約 0.54~0.55
B	1	128	0.4	約 0.97	約 0.53
C	2	128+128	0 + 0	約 0.99~1	約 0.47~0.49
D	2	128+128	0.25 + 0.25	約 0.86	約 0.50~0.55
E	2	128+128	0.4 + 0.4	約 0.81	約 0.51

  若 neuron 數加大 B 結果比 A 好，推測 B 在訓練過程 neuron 太少，故結果不佳。

- 1 層 layer、神經元數為 500、epoch = 50，各 lambda 的結果比較

neuron	regularization	dropout	train acc	validation acc
500	0.01	0.25	0.988	0.534
500	0.01	0.4	0.97	0.48~0.52
500	0.03	0.4	0.854	0.544
500	0.1	0.4	0.78	0.50

- 若 1~2 層 layer，則 Activation 均為 sigmoid 結果較佳；若超過 3 層，則不超過 3 層 sigmoid，且前幾層改為 relu 結果較佳，否則會有神經元輸出的差遞減的問題。

2. Semi-Supervised: Self-Training

Method -- 將 unlabel data 輸入第一部分訓練好的 model，計算 unlabel 在十種類別的機率及對應的 entropy，若 entropy 小於 threshold（約 0.1~0.5），則加入對應的 label，並重複訓練此過程 1~4 次，最終約訓練 15000~20000 筆資料。

Performance

- 以下條列較具代表性的機率對應的 entropy，並照 entropy 大小排列：

entropy	P(0)	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	P(7)	P(8)	P(9)
0.163	0.97	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.02	0.00
0.453	0.00	0.87	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.10
0.658	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.79	0.00	0.16
0.923	0.01	0.00	0.57	0.00	0.38	0.00	0.03	0.00	0.01	0.00
1.668	0.19	0.29	0.17	0.06	0.01	0.00	0.00	0.02	0.26	0.00

- 目前傳到 kaggle 上的 public 分數約為 0.45~0.51 不等，並沒有比只使用 label 高。

3. Semi-Supervised: Autoencoder

Method

- 此部分先把所有資料除 255.0，autoencoder 訓練過程的 lost 才會穩定下降。
- Model 和對應的大小：

  InputLayer (3, 32, 32) -> Convolution (16, 32, 32) -> MaxPooling (16, 16, 16)
  -> Convolution (16, 16, 16) -> MaxPooling (16, 4, 4) -> Convolution (3, 4, 4)
  -> Convolution (16, 4, 4) -> UpSampling (16, 16, 16)
  -> Convolution (16, 16, 16) -> UpSampling (16, 32, 32) -> Convolution (3, 32, 32)
- encode 後大小為 (16, 4, 4)，經過 flatter() 變成 256 維，將 10 種類別各自的 500 筆 256 維的 code 取平均作為 10 個 cluster 的中心點。
- 將 unlabel data 通過同樣的 encode 過程，與 10 個中心點計算相似度，取出最近並小於某 threshold 的 unlabel 與原資料合併，最終約有 20000~25000 筆資料。

Performance -- 在 kaggle 的 public 分數上，結果約為 0.4，又更差，推測可能是 autoencoder 的 model 沒有選擇恰當（如層數或 filter 數目），因為在 autoencoder 的過程訓練極慢（encode/decode 各兩層，1min/1epoch）沒辦法測試較多種組合。

4. Compare and Analyze Result

	Supervised	Self-Training	Autoencoder
train acc	~1	0.90~0.94	0.6~0.8
validation acc	0.558	0.50~0.53	0.379~0.42
public set	0.543	0.511	0.382

- 兩種 semi-supervised 的結果都沒有比只訓練 5000 筆好，有多次檢查程式碼應該沒有疏忽打錯的地方，所以可能是 unlabel 分群不準確，導致重複訓練時效果變差。
- Self-Training 與 auto-encoder 相較之下，每一次的訓練時間短很多，並且 Self-Training 可看出 unlabel 資料通過 model 後得到的十項機率，auto-encoder 可看出離十群中心點的相似度。我個人認為前者比較能直接觀察、設定恰當 threshold，而後者則需要經過實際運行程式，才能判斷出適合的 threshold。