Machine Learning 2016 Fall Homework 3

b03902035 黃兆緯

1. Supervised Learning
   (1) Models
   由於是圖像分類問題，因此都使用了現在最流行的 convolutional layer 來建立模型，在參考了一些論文之後，使用 Keras 建立了以下三個 architecture
   (a) Simple CNN
   參考 Keras 的 cifar10 example 的架構，增加了 filter 數量：
   $input\ -(128\,nC3-128\,nC3-MP2-Drop(0.25))_3-1024-512-10-\ output$
   其中 C3 為 3·3 的 convolutional layer，MP2 為 2·2 的 MaxPooling，1024-512-10 為 fully-connected layer，$(...)_3$ 是同樣的 block 重複三次，而 128n 就是依照重複的次數成長(128→256→384)。
   (b) VGG-style CNN
   參考 VGG 組織發表的”Very Deep Convolutional Networks for Large-Scale Image Recognition”文章，增加了 filter 數量，由於圖片較小，因此減少了一些 layer：
   $input\ -(192\,nC3-192\,nC3-MP2-Drop(0.25))_2-$
   $(768\,C3)_4-MP2-Drop(0.25)-1024-512-10-\ output$
   (c) All-CNN
   參考”Striving for Simplicity: The All Convolutional Net” 文章，一樣調整了 filter 數量：
   $input\ -(200\,nC3-200\,nC3-MP2-Drop(0.1\,n))_5-1200\,C2-1200\,C1-10-output$
   這個 model 的特點是全部都是 convolutional layer，只有最後一層 10 個 neuron 的 fully-connected layer
   (2) Data Augmentation
   使用 Keras 的 ImageDataGenerator 來隨機變換 input image，用到的變換包括上下左右偏移和水平翻轉，使用 Data augmentation 讓正確率提升大約 5% (0.5→ 0.55)
   (3) Model Comparison

	Simple CNN	VGG-style CNN	All-CNN
# of layers	12	14	18
# of parameters	10.07M	34.03M	21.48M
Training time per epoch (sec)	22s	90s	75s
Accuracy	~0.78	~0.78	~0.79

   由於 VGG-style CNN 和 All-CNN 的 training time 都太久了，因此只各訓練了一次作為 ensembling 的 voters。

2. Semi-supervised Learning – Self-training
   (1) 方法
   有 5000 筆 labeled data 和 45000 筆 unlabeled data，先做 supervised learning 後，取 unlabeled data 中 confidence 較高的加入 labeled data 中，方法如下：
   (a) $X_l = labeled\ data, X_u = unlabeled\ data$
   (b) $supervised\ learning:\ \ f(X_l)$
   (c) $prediction_u = f(X_u)$
   (d) 計算 $k = 0.1 \cdot min(\#\ of\ labels\ \text{in}\ prediction\ \ for\ \ each\ \ class)$
   (e) 取各個 `class` 中 `confidence` 最大的前 k 個，加入成為新的 labeled data
   (f) 重複 a-e 數次

(2) Performance

	# of samples	accuracy
Base model	5000	0.78
Self-training iteration # 1	9000	0.795
Iteration # 2	12700	0.805
Iteration # 3	16000	0.815
Iteration # 5	22000	0.84

3. Semi-supervised Learning – Autoencoder

(1) Model

參考 Keras example 中的 convolutional autoencoder:

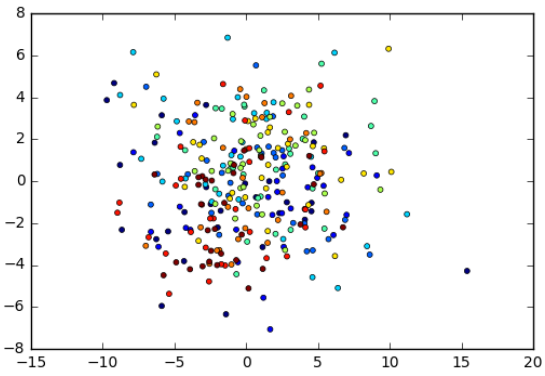
$input - (16C3 - MP2)_3 - encoded - (16C3 - US2)_3 - output$

US 為 UpSampling layer。

取出 256 維 feature 後，再輸入 DNN 用 labeled data 做 supervised learning，DNN:

$input - encoder - 1024 - 512 - 10 - output$

(2) Performance



上圖是 labeled data 經過 encoder 得到 256 維 feature，再經過 PCA 得到的二維分佈，不同顏色代表不同 class 的 data，可以看到分佈都滿平均的，也就是 encoder 沒有將圖片 encode 成明顯有用的 feature。

輸入 DNN 用 labeled data 做 supervised learning 後，得到的 accuracy 大約是 0.45，完全比不上單純 supervised learning 的結果。

4. Analysis

(1) # of parameters

經過幾次實驗，我發現隨著參數量(filter 數量)的提昇，performance 也會有提昇，但也有上限。下表為實驗結果:

model	CNN 32n	CNN 128n	CNN 384n	VGG 96n	VGG 192n
Training time	5s	22s	120s	35s	90s
accuracy	0.71	0.78	0.79	0.72	0.785

(2) Batch Normalization

參考”Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift” 文章，文中提到 Batch Normalization 可以加速 converge，在每個 convolutional

layer 和 activation 之間加入 BN layer，Simple CNN 的 accuracy 從 0.55 上升到了 0.7。有可能是原本訓練不夠久，還沒有 converge，加入 BN 之後比較快到達好的參數。

(3) Learning Rate

這次作業的模型對於 learning rate 非常敏感，learning rate 差一個數量級就可能完全無法訓練，例如 0.001 太大，而 0.0001 太小，造成 converge 需要很久，因此最後取 0.0003 來訓練。

(4) Temporal Ensembling

參考 NVIDIA 發表的”Temporal Ensembling for Semi-Supervised Learning”，在預測 unlabeled data 時，使用上一個 iteration 的最後 10 個 epoch 的 model 來投票，這樣可以讓 model 的 accuracy 更穩定(training 時起伏很大)，也可以提高正確率(大約 2%)