Wrocław, 23.06.2017
Bartosz Rybicki

Data Scientist Exercise - Raport

W celu rozwiązania zadania wykorzystałem bibliteki keras oraz scikit-learn dostępne w Python'ie. Poniżej znajduje się opis sposóbu w jaki rozwiązałem kolejne podzadania.

Uwaga ogólna: jeśli chcą państwo faktycznie uruchomić przygotowany przeze mnie program to proszę wykonywać skrytpy według kolejności określonej przez zadania.

Niestety nie posiadam dostępu do GPU z dostępnym CUDA, więc zgodnie z uwagą w poleceniu we wszystkich kolejnych zadaniach ograniczyłem się do 10% danych dostępnych w zbiorze CIFAR-10.

Zadanie pierwsze i drugie. Rozwiązanie ogranicza się do pobrania i rozpakowania danych, wybrania odpowiednich rysunków oraz wyświetlenia ich w odpowiedni sposób. Sposób przeglądania etykiet (tj. „labels") w celu wybrania dziesięciu obrazków z każdej kategorii został zaimplementowany tak by każdą etykietę oglądać tylko raz. Efekt mojej pracy jest widoczny poniżej.

Zadanie trzecie. W ramach tego zadania przygotowałem dwa benchmark'i. Oba jako wejście dostają macierz w której każdy wiersz jest „opisem" jednego obrazka ze zbioru CIFAR-10.

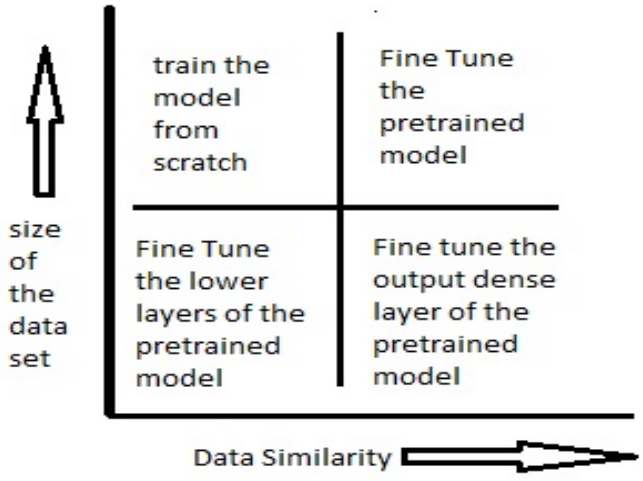
Pierwszy benchmark redukuje liczbę cech do 32 poprzez zastosowanie NMF które jest następnie podane jako wejście do SVM z jądrem liniowym oraz kara równą 250. Tak zbudowany klasyfikator przy 10-krotnej kroswalidacji daje dokładność 0.35 z odchyleniem równym 0.02.

Drugi benchmark wyłuskuje deskryptory HoG, które są następnie przekazane do PCA, który redukuje liczbę cech do 32. Tak zmodyfikowane dane są przekazane do SVM z jądrem liniowym raz karą 60. Ten klasyfikator przy 10-krotnej kroswalidacji daje dokładność 0.38 z odchyleniem równym 0.01.

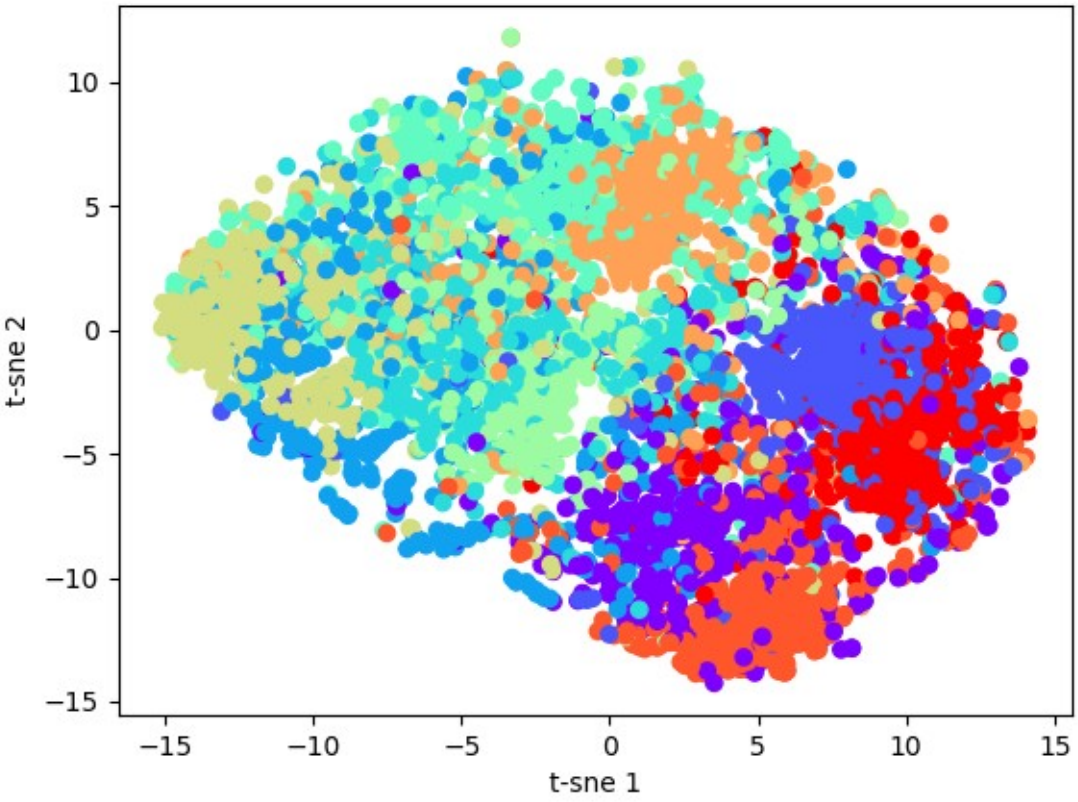
Drugie z zaproponowanch podejść daje lepsze wyniki, wiec zostaje wybrane jako właściwy benchmark.

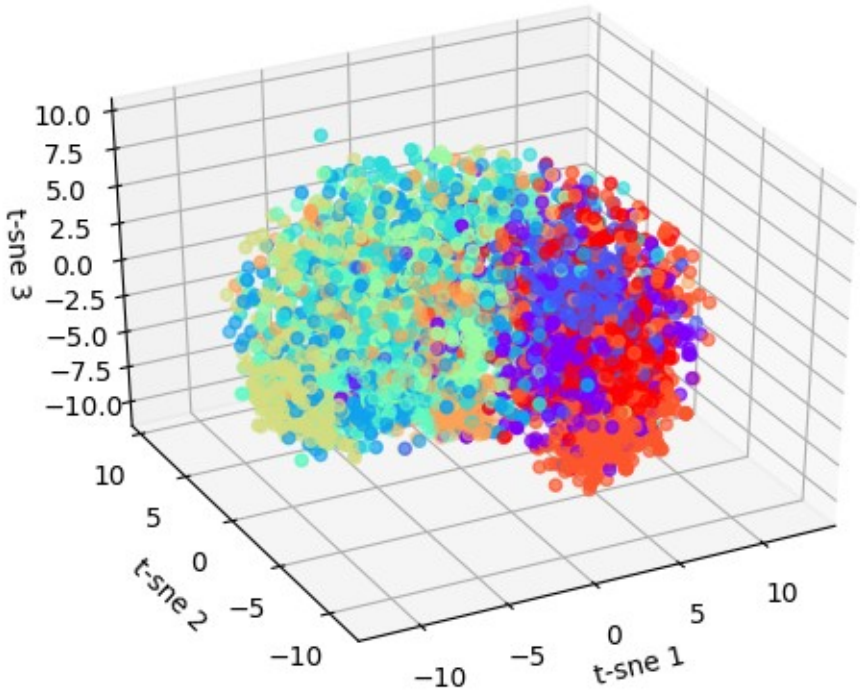
Zadanie czwarte. Początkowo wykorzystywałem biblitoekę TFLearn, która ma wyjątkowo czytelny interface, ale niestety występują w niej (znane) problemy z pamięcią co rzutowało na wydajność. Dlatego postanowiłem wkorzystać bibliotekę Keras, która sprawowała się bez zarzutu.

Wybraną przeze mnie siecią jest „Inception V3". W wyniku kilku prób oraz lektury artykułów w internecie zdecydowałem się na odczytanie „CNN codes" z warstwy pool_3 (w implementacji dostępnej w Keras ta wartstwa nazwana jest avg_pool) znajdującej się bezpośrednio przed warstwą softmax. Zrobiłem tak ponieważ zbiór danych był mały oraz podobny to zbioru na którym sieć była trenowana.



Zadanie piąte. W celu wizualizacji „CNN codes" wykorzystałem metodę t-SNE, która zredukowała wektory w dwóch oraz trzech wymiarów. Metoda t-SNE ma tą własność że stara się zachowywać odegłości między punktami więc nadaje się idealnie do tego typu zadań. Wyniki mojej pracy są widoczne poniżej.


t-sne 2
t-sne 1
10
5
0
−5
−10
−15
−15
−10
−5
0
5
10
15


t-sne 3
t-sne 2
t-sne 1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
10
5
0
−5
−10
−10
−5
0
5
10

Zadanie szóste oraz siódme. Zgodnie z poleceniem potraktowałem „CNN codes” jako wejście do klasyfikatora SVM. Dane zostały podzielone w sposób losowy na zbiór treningowy oraz testowy w stosunku 9:1. Krotności obserwacji z poszczególnych klasy w obu zbiorach są zwizualizowane poniżej.

Po przetestowaniu różnych zbiorów parametrów klasyfikatora SVM zdecydowałem się na jądro liniowe, z karą 0.01, co w efekcie dało mi klasyfikator o dokładności 0.76 na zbiorze testowym (dwókrotnie lepszy wynik niż benchmark). Dokładne wyniki dla precision, recall oraz F1-score sa dostępne w poniższej tabeli. Na jakość uzyskanego klasyfikatora wpływ ma ilość wykorzystanych danych uczących (przypominam że wykorzystuje 10% zbioru CIFAR-10 i zapewnie dlatego nie uzyskałem skuteczności na poziomie 0.87).

	Precision	Recall	F1-score	Count
Airplane	0.78	0.84	0.81	58
Automobile	0.83	0.81	0.78	62
Bird	0.78	0.79	0.78	62
Cat	0.52	0.65	0.58	52
Deer	0.72	0.67	0.70	70
Dog	0.65	0.63	0.64	57
Frog	0.80	0.74	0.77	72
Horse	0.84	0.76	0.80	49
Ship	0.85	0.82	0.84	62
Truck	0.85	0.86	0.85	64
Avg / Total	0.76	0.76	0.76	600

F1-score ważony według krotności poszczególnych klas ma wartość 0.76.

„Confision matrix” (macierz zamieszania ;-) dostępna jest poniżej:

49	0	2	0	1	1	0	1	3	1
0	44	0	1	0	2	0	0	1	6
1	0	49	2	4	2	3	0	0	1
1	0	2	34	3	8	2	1	1	0
0	0	5	5	47	2	2	5	2	2
1	1	1	10	4	36	4	0	0	0
1	0	4	8	2	3	53	0	1	0
2	0	0	5	3	1	1	37	0	0
7	2	0	0	1	0	1	0	51	0
1	6	0	1	0	0	0	0	1	55

Zadanie ósme (bonusowe). Zdecydowałem sie na wykorzystanie modelu z rodziny „ensemble”, konkretnie RandomForest. Niestety uzyskane w ten sposób wyniki są zdominowane przez te uzyskane przy pomocy SVM.

	Precision	Recall	F1-score	Count
Airplane	0.74	0.74	0.64	58
Automobile	0.78	0.83	0.80	62
Bird	0.73	0.58	0.65	62
Cat	0.46	0.58	0.51	52
Deer	0.69	0.60	0.64	70
Dog	0.64	0.67	0.66	57
Frog	0.71	0.74	0.72	72
Horse	0.71	0.73	0.72	49
Ship	0.85	0.85	0.85	62
Truck	0.82	0.80	0.81	64
Avg / Total	0.72	0.71	0.71	600

43	0	4	2	2	0	0	1	5	1
0	45	0	1	0	0	1	0	1	6
4	0	36	5	5	6	5	1	0	0
2	1	2	30	6	7	3	0	0	1
0	2	2	2	42	1	8	10	1	2
1	1	0	10	2	38	3	2	0	0
1	0	3	10	2	3	53	0	0	0
1	0	1	4	2	3	1	36	1	0
6	0	1	0	0	0	1	0	53	1
0	9	0	1	0	1	0	1	1	51