Informatyka, studia dzienne, mgr II st. semestr II

Rozpoznawanie obrazów **2011/2012**

Prowadzący: dr inż. Bartłomiej Stasiak poniedziałek, 8:15

Data oddania: _____ Ocena: _____

Michał Janiszewski 169485

Zadanie 1: przestrzeń cech, ekstrakcja cech, klasyfikacja, metody minimalnoodległościowe (metoda k-NN).*

1. Cel zadania

Celem zadania było stworzenie uniwersalnego szkieletu aplikacji dokonującej klasyfikacji obiektów. Aplikacja ma być niezależna od rodzaju danych i umożliwiać klasyfikację metodą k-NN za pomocą wybranej metryki.

Działanie aplikacji powinno zostać zaprezentowane na przykładzie rozpoznawania pisanych odręcznie cyfr arabskich.

2. Wprowadzenie

2.1. Klasyfikacja

Zadanie klasyfikacji polega na rozpoznaniu nowych, nieznanych obiektów w celu przypisania im etykiety klasy do której należą, na podstawie znajomości cech badanego obiektu oraz pewnego zbioru obiektów uczących o znanych cechach.

Klasyfikacji dokonuje się w przedstawionych poniżej etapach [2]:

1. stworzenie klas na podstawie zadanego zbioru obiektów,
2. określenie charakterystyk klas,
3. określenie podobieństwa nowych obiektów do klas.

* SVN: `https://serce.ics.p.lodz.pl/svn/labs/sise/mp_pt0830/jankit/genetyk`

Ponieważ w wyznaczonym zadaniu klasy obiektów są już utworzone, skupimy się na etapach 2 i 3.

Etap 2, czyli ekstrakcja cech, polega na wyodrębnieniu dla każdego z obiektów n cech, które tworzą n-wymiarową przestrzeń cech. Podczas wyznaczania przestrzeni cech należy kierować się zasadą Brawermanna, która mówi, że obiekty jednej klasy powinny tworzyć skupiska maksymalnie zwarte i możliwie najbardziej oddalone od skupisk obiektów innych klas.

Etap 3 można zrealizować poprzez znalezienie odległości badanego obiektu do obiektów klas uczących zgodnie z pewną ustaloną metryką.

Metryką ρ w przestrzeni $X = R^n$ nazywamy odwzorowanie, które spełnia następujące założenia:
1. $\forall \overrightarrow{x}, \overrightarrow{y} \in X : \rho(\overrightarrow{x}, \overrightarrow{y}) \in R, \rho(\overrightarrow{x}, \overrightarrow{y}) \geq 0$
2. $\forall \overrightarrow{x} \in X : \rho(\overrightarrow{x}, \overrightarrow{x}) = 0$
3. $\forall \overrightarrow{x}, \overrightarrow{y} \in X : \rho(\overrightarrow{x}, \overrightarrow{y}) = \rho(\overrightarrow{y}, \overrightarrow{x})$
4. $\forall \overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z} \in X : \rho(\overrightarrow{x}, \overrightarrow{y}) + \rho(\overrightarrow{y}, \overrightarrow{z}) \geq \rho(\overrightarrow{x}, \overrightarrow{z})$

2.2. Metoda k-NN

Metoda *k-NN* (ang. *k nearest neighbours* – k najbliższych sąsiadów) jest metodą minimalnoodległościową, co oznacza, że nadanie etykiety klasy opierać będzie się o znalezienie k takich elementów, od których odległość od badanego obiektu w przestrzeni cech X mierzona metryką ρ będzie najmniejsza. W tym celu należy znaleźć odległości od badanego elementu ze zbioru testowego do każdego z elementów zbioru uczącego, a następnie wybrać k elementów z najmniejszą odległością. Badanemu obiektowi przypisana zostanie etykieta klasy, która posiadała najwięcej reprezentantów wśród znalezionych k najbliższych sąsiadów.

Wylosowany wariant zadania opierał się na metryce Minkokwskiego z parametrem $t = 3$:

$$\rho_m^t(\overrightarrow{x}, \overrightarrow{y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^t \right)^{\frac{1}{t}} \tag{1}$$

3. Implementacja

Implementacja została zrealizowana w języku C++ z wykorzystaniem środowiska Qt.

Przygotowany został zestaw aplikacji:
1. `extractor` – aplikacja dokonująca ekstrakcji cech z obrazów,
2. `classifier` – aplikacja dokonująca klasyfikacji metodą k-NN wektorów cech dostarczonych przez `extractor`.

3.1. extractor

Zadaniem programu `extractor` jest przetworzenie danych znajdujących się w plikach pobranych z bazy MNIST na wektory cech każdego z elementów. Za pomocą klasy `ArchiveExtractor` wczytywane są archiwa zawierające obrazki oraz etykiety, następnie wykorzystując interfejs (klasę czysto

abstrakcyjną) `FeatureExtractorInterface`, zainicjowany uprzednio fabryką `FeatureExtractorFactory`, dokonywana jest ekstrakcja cech, które zapisywane są strumieniem tekstowym do podanego w wywołaniu programu pliku.

Ekstraktory cech implementujące interfejs `FeatureExtractorInterface` opisane są w sekcji 4.1.

3.2. classifier

Program ten za pomocą klasy `FeatureImporter` wczytuje z podanych w argumentach programu plików wektory cech elementów zbioru testowego oraz uczącego. Wyekstrahowane cechy przekazywane są do klasy implementującej interfejs `ClassifierInterface` dokonującej klasyfikacji.

Aktualnie istnieje tylko jedna klasa implementująca powyższy interfejs, jest to `CpuClassifier`.

4. Materiały i metody

W zadaniu wykorzystany został zestaw przygotowanych pisanych odręcznie cyfr arabskich dostępnych pod adresem `http://yann.lecun.com/exdb/mnist/`.

4.1. Ekstraktory cech

Do ekstrakcji cech wykorzystano następujące ekstraktory implementujące interfejs `FeatureExtractorInterface`:

— `LeftRightProfile` – ekstraktor ten tworzy profil lewej oraz prawej strony obrazka, czyli wyznacza numery pikseli w poziomej linii, w której po raz pierwszy (z lewej) i po raz ostatni (z prawej) przekroczona jest wartość progu. W celu niwelacji przesunięcia całej cyfry wewnątrz obrazka, wartości są wyrównywane poprzez „dosunięcie" obrazka do odpowiedniej strony.

— `Crossing` – ekstraktor ten określa, ile razy w każdej poziomej i pionowej linii następuje przejście przez próg.

— `Projection` – ekstraktor ten zlicza w każdej linii pionowej i poziomej ilość pikseli przekraczających próg.

— `Zones` – ekstraktor ten wyznacza pewną ilość stref zarówno w pionie jak i poziomie, a następnie zlicza ilość pikseli przekraczających próg w każdej z nich.

Wartość progu dla wszystkich ekstraktorów wynosiła 127.

4.2. Cache-owanie sąsiadów

Jak łatwo zauważyć z opisu metody k-NN, wybranie k_i najbliższych sąsiadów powoduje jednocześnie możliwość dokonania klasyfikacji dla k_j, gdzie $0 < k_j \leq k_i$ bez konieczności ponownego wyznaczania odległości pomiędzy poszczególnymi elementami.

Opisana powyżej cecha została zaimplementowana w programie dokonującym klasyfikacji, dzięki czemu podając maksymalną wartość parametru k w

wyniku działania otrzymuje się też wyniki dla wszystkich mniejszych wartości k.

5. Wyniki

Rysunek 1 pokazuje skuteczność poszczególnych klasyfikatorów w zależności od parametru k. Rysunek 2 pokazuje czas przetwarzania całego zbioru danych dla poszczególnych klasyfikatorów w zależności od parametru k.

Tabele 1, 2, 3 oraz 4 prezentują macierz pomyłek dla każdego ekstraktora dla parametru $k = 1$[1]. Należy interpretować je w następujący sposób: wartość w komórce (i, j), gdzie i to numer kolumny, zaś j – wiersza, oznacza ile (procentowo) razy klasa z etykietą z kolumny i została uznana za klasę z etykietą z wiersza j.

6. Dyskusja

Najskuteczniejszym ekstraktorem cech okazał się `LeftRightProfile`, który uzyskał wynik 91,54% dla $k = 1$ i dla wszystkich sprawdzonych wartości k był dokładniejszy niż pozostałe ekstraktory. Oznacza to, że ekstraktor ten najlepiej wypełnia regułę Brawermanna: najskuteczniej separuje od siebie zwarte skupiska klas.

Wraz ze wzrostem parametru k, skuteczność klasyfikacji maleje, jednak najbardziej jest to widoczne w ekstraktorze `Projection`, najmniej w przypadku `LeftRightProfile`. Dzieje się tak, ponieważ w otoczeniu banadych obiektów znajduje się coraz więcej elementów z etykietami innych klas, co powoduje zaburzenia klasyfikacji.

Zaprezentowany na rysunku 2 czas przetwarzania dla różnych wartości k jest stały. Wynika to z zastosowania struktury imitującej kolejkę do przetrzymywania najbliższych sąsiadów, dzięki czemu nie jest wymagane jej każdorazowe sortowanie; także przeglądanie zawartych w niej elementów ograniczone jest do minimum.

7. Wnioski

Analizując macierze pomyłek można zauważyć, że dla pewnych ekstraktorów większość pomyłek zachodzi dla pewnych określonych par. Aby temu zaradzić można by, po zapoznaniu się z wynikami działania programu, wprowadzić dodatkowy przebieg klasyfikacji innym ekstraktorem cech, posiadającym dla danej pary małą wartość pomyłek.

Jak widać, opisywane w sekcji 4.2 usprawnienie nie wpływa znacząco na czas przetwarzania całego zbioru danych, pozwala natomiast na uzyskanie wszystkich wartości już w jednym przebiegu programu.

[1] Dane dla wszystkich k znajdują się na repozytorium, gdyż w sprawozdaniu zajęłyby zbyt dużo miejsca.

Tabela 1. Macierz wyników ekstraktora `Projection` dla $k = 1$

	0	1	2	3	4	5	6	7	8	9
0	90.97	0.615	0.501	0	0.216	0.132	0.101	0.578	1.151	0
1	0.28	85.99	0.501	0	0	0.528	0.202	0	0.460	0
2	1.057	0.461	84.97	3.229	1.082	5.548	5.152	1.061	2.071	0.274
3	0.672	1.463	3.707	78.41	0.216	10.96	0.303	2.797	3.913	1.737
4	0.480	1.078	0.601	0	89.83	0	0.505	0.675	0.460	10.15
5	1.633	0.769	6.413	11.81	0.216	76.22	0.909	2.989	5.063	1.92
6	0.768	1.54	1.202	0.100	0.974	0.660	90.71	0	0.460	0.091
7	0.192	2.002	1.202	1.615	0.974	0.924	0.101	87.85	0.460	3.656
8	3.458	4.619	0.801	3.532	0.865	3.963	1.919	2.025	84.81	1.828
9	0.480	1.463	0.1002	1.312	5.628	1.057	0.101	2.025	1.151	80.35

Tabela 2. Macierz wyników ekstraktora `Crossing` dla $k = 1$

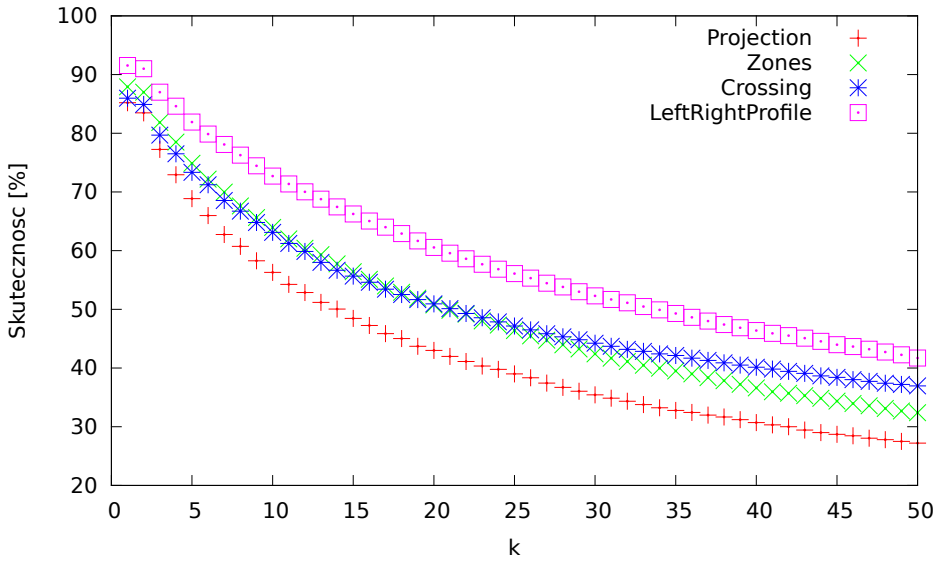
	0	1	2	3	4	5	6	7	8	9
0	91.83	0	0	0.106	0.426	0.212	0.401	0.091	0.107	0.103
1	0	96.44	0.712	0.213	0.426	0.106	0.501	0.274	0	0.206
2	1.141	0.260	76.6	8.094	0.746	10.08	4.213	0.731	3.644	0.206
3	0.190	0	8.444	80.51	0.213	11.89	0.200	2.102	1.715	1.443
4	1.996	1.128	0.610	0.106	91.15	0.318	0.401	2.011	1.179	4.742
5	0.855	0.520	8.647	7.455	0.426	70.59	1.906	1.28	1.179	0.927
6	0.570	0.434	1.424	0.319	0.533	1.274	90.87	0	0.643	0.103
7	0.285	0.434	1.119	1.065	1.066	1.911	0	84.1	0.643	4.639
8	2.091	0.520	2.136	1.171	1.386	2.017	1.505	1.097	89.5	2.062
9	1.046	0.260	0.305	0.958	3.625	1.592	0	8.318	1.393	85.57

Tabela 3. Macierz wyników ekstraktora `LeftRightProfile` dla $k = 1$

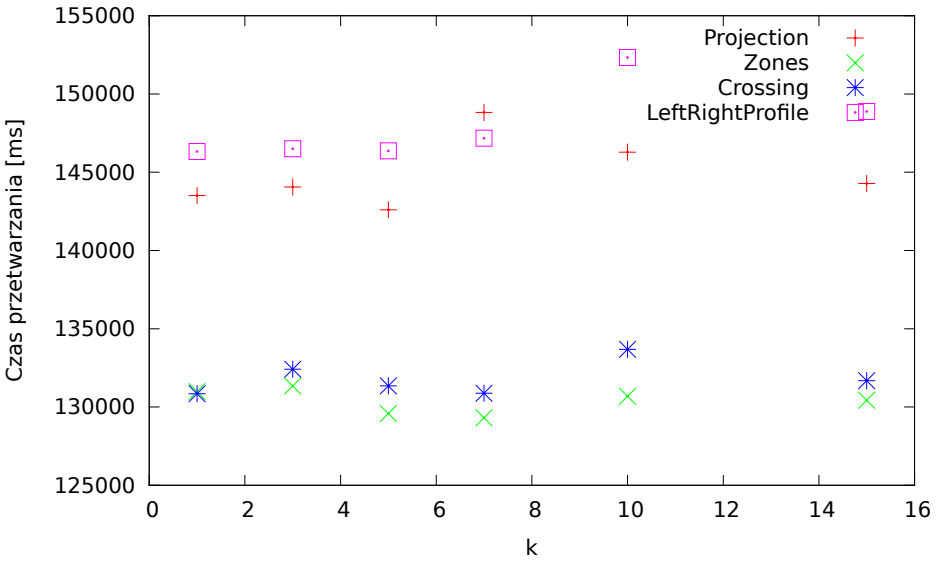
	0	1	2	3	4	5	6	7	8	9
0	92.43	0.261	0.3	0.101	0.108	0	0.514	0.306	0.611	0.446
1	0.194	96.08	0.9	0.101	0.325	0.233	0.205	0	1.222	0.089
2	0.679	0.348	94.6	2.121	0.216	0.467	0.514	1.021	2.953	0.356
3	0.194	0.609	1.6	93.23	0.108	1.754	0	1.634	1.731	1.16
4	0.872	0.348	0.4	0	91.76	0.233	1.029	1.634	1.324	6.958
5	0.291	0.174	0.7	2.525	0.325	95.2	0.308	0.408	1.935	1.07
6	0.872	0.348	0.4	0	0.216	0.350	95.99	0	0.305	0
7	0	0.696	0.8	0.909	1.844	0.233	0	89.58	0.611	9.01
8	4.171	0.696	0.2	0.202	1.193	0.701	1.44	0.817	88.19	1.249
9	0.291	0.435	0.1	0.808	3.905	0.818	0	4.597	1.12	79.66

Tabela 4. Macierz wyników ekstraktora `Zones` dla $k = 1$

	0	1	2	3	4	5	6	7	8	9
0	93.69	0.165	0.602	0.406	0.103	0.572	0.717	0.205	1.554	0.189
1	0.100	91.39	0.200	0.304	0.412	0.114	0.410	0	1.554	0.094
2	0.900	0.496	94.08	1.827	0.928	0.228	0.820	1.951	2.28	0.189
3	1.301	0.993	1.606	84.37	0.309	7.323	0	1.745	4.145	1.327
4	0.100	0.496	0.100	0.101	85.86	0.572	2.256	1.54	0.621	8.815
5	0.700	0.413	0.401	6.396	0.928	85.7	1.231	0.308	3.316	0.758
6	0.500	0.993	0.100	0.101	1.032	1.373	92.92	0.102	0.829	0.189
7	0	0.993	1.807	0.913	3.199	0.457	0.102	88.3	0.829	8.057
8	2.302	3.063	0.602	3.655	1.445	3.089	1.538	0.308	82.9	1.232
9	0.400	0.993	0.502	1.929	5.779	0.572	0	5.544	1.969	79.15



Rysunek 1. Skuteczność ekstraktorów względem wartości k



Rysunek 2. Czas przetwarzania względem wartości k

Literatura

[1] *Wykład*, Mykhaylo Yatsymirskyy, Łódź, 2011

[2] *Metody wyszukiwania i klasyfikacji informacji*, Mirosław Dąbrowski, Krystyna Laus-Mączyńska, Wydawnictwa Naukowo-Techniczne, Warszawa 1978